VŠB – Technická univerzita Ostrava
Fakulta elektrotechniky a informatiky
Katedra informatiky

Data mining v oblasti e-learningových systémů

Data Mining in e-learning systems

2017 Michal Popek

VŠB - Technická univerzita Ostrava
Fakulta elektrotechniky a informatiky
Katedra informatiky

Zadání bakalářské práce

Student: **Michal Popek**

Studijní program: B2647 Informační a komunikační technologie

Studijní obor: 2612R025 Informatika a výpočetní technika

Téma: Data mining v oblasti e-learningových systémů
Data Mining in E-learning Systems

Jazyk vypracování: čeština

Zásady pro vypracování:

Cílem práce je prostudovat oblast data mining v e-learningových systémech, vybrat a implementovat data miningove metody vhodné pro oblast e-learningu.

1. Prostudovat možnosti data miningu (DM) v LMS.
2. Seznámit se s některým LMS (e-Logika nebo Moodle) a s DM metodami v něm aplikovanými.
3. Implementovat další DM metody do vybraného systému.
4. Analýza výsledků DM v LMS, doporučení pro uživatele.
5. Přehledná reprezentace získaných výsledků.

Seznam doporučené odborné literatury:

[1] Šarmanová, J.. Metody analýzy dat, VŠB-TU Ostrava, 2012.
[2] Peña-Ayala, A. Educational data mining: A survey and a data mining-based analysis of recent works. Expert Systems with Applications 41, 1432-1462,2014.
[3] Jindal, R. & Borah, M.D. A Survey on Educational Data Mining and Research Trends. International Journal of Database Management Systems, 53-73, 2013.

Formální náležitosti a rozsah bakalářské práce stanoví pokyny pro vypracování zveřejněné na webových stránkách fakulty.

Vedoucí bakalářské práce: **Mgr. Pavla Dráždilová, Ph.D.**

Datum zadání: 01.09.2014

Datum odevzdání: 14.07.2017

doc. Ing. Jan Platoš, Ph.D.
vedoucí katedry

prof. RNDr. Václav Snášel, CSc.
děkan fakulty

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně. Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

V Ostravě 14. července 2017

Souhlasím se zveřejněním této bakalářské práce dle požadavků čl. 26, odst. 9 *Studijního a zkušebního řádu pro studium v bakalářských programech VŠB-TU Ostrava.*

V Ostravě 14. července 2017 .....

Rád bych poděkoval paní Mgr. Pavle Dráždilové, PhD. za konzultace, rady a vstřícný přístup, který dopomohl k dokončení této bakalářské práce. Dále bych také rád poděkoval panu Ing. Martinu Prokešovi za konzultaci k systému eLogika a poskytnutí dat.

Abstrakt

Tématem této bakalářské práce je dolování dat v e-learningových systémech. Cílem je analyzovat, jaká data se ukládají ve vybraném systému eLogika a posoudit možností dolování dat z tohoto systémů. Dále je součástí je i vytvoření aplikace pro analýzu dat pocházející ze systému eLogika. V první části této práce je uvedení do oblasti dolování dat v e-learningu a popis vybraných dolovacích metod. Následuje vývoj aplikace a příklady použití.

Klíčová slova: Data mining, e-learningové systémy, eLogika, předzpracování

Abstract

The subject of this thesis is data mining in e-learning systems. The goal is to analyze what data is stored in the selected eLogika system and to evaluate the data mining capabilities of the systems. It also includes the creation of an application for data analysis originating from the eLogika system. In the first part of this thesis is introduction into the field of data minig in e-learning and description of selected data mining methods. Following are the development of the application and examples of use.

Key Words: Data mining, e-learning systéms, eLogika, pre-procesing

Obsah

Seznam použitých zkratek a symbolů

IEDMS	–	International educational data mining society
ER	–	Entity Relationship diagram
LMS	–	Learning manegment system
T-SQL	–	Transact Structured Query Language
WPF	–	Windows Presentation Foundation
XML	–	eXtensible Markup Language
MPZZ	–	Matematika pro zpracování znalostí

Seznam obrázků

Seznam tabulek

Seznam výpisů zdrojového kódu

1 Úvod

Tématem bakalářské práce je seznámení s dolováním dat v oblasti e-learningových systémů se zaměřením na určitý e-learningový systém. Prozkoumat uložená data a na ně aplikovat vlastnoručně naimplementované dolovací metody. Následně provést experimenty a výsledky přehledně reprezentovat s vhodným popisem a doporučením.

Na začátku se seznámíme s podstatou e-learningových systému a procesem dolování dat v těchto systémech.

V následující kapitole se zaměříme na vybrané dolovací metody. Jsou zde popsány základní statistické metody, shlukování a asociační pravidla. V případě shlukování to jsou aglomerativní metody jako například metoda nejvzdálenějšího souseda, nejbližšího souseda, průměrné vazby, centroidu nebo Wardova metoda. Co se týče asociačních pravidel je použita metoda Fp-Growth pro vyhledávání frekventovaných množin.

Ve čtvrté kapitole zanalyzujeme data produkované systémem eLogika, které mohou být z testování studentů nebo logování jejich aktivit uvnitř systému.

V následující kapitole, se zaměříme na analýzu požadavku pro výslednou aplikaci. Bude zde také popsán případ užití a jednotlivé scénáře.

Šestá kapitola se bude zabývat implementační stránkou aplikace. Bude zde popsána struktura vlastního datového úložiště se strukturou vytvořených datasetů. Dále zde bude popsána struktura výsledné aplikace. To znamená implementace přístupu k datům, dolovacích algoritmů a vlastních reprezentativních prvků jako je například dendrogram.

Nakonec budou popsány experimenty, které se prováděly na vytvořených datasetech s přehlednou reprezentací výsledku.

Toto téma je velice zajímavé z pohledu aktuálnosti. V dnešní době informačních technologii se výuka postupně schyluje k automatizaci pomocí výukových systémů jako je například Moodle nebo eLogika. Tyto systémy produkují velké množství skrytých užitečných informací, které mohou pomoci zefektivnit výuku s maximální odezvou.

2 Dolování dat v oblasti e-learningu

V posledních letech se v hojném počtu začaly vytvářet e-learningové systémy pro snadnější řízení výuky (viz. kapitola 2.1). Tyto systémy produkují velké množství dat o průběhu vzdělávání, které je možné analyzovat. Pokud bychom data procházeli jednotlivě, tak by jejich analýza byla značně náročná, a to především kvůli jejich rozsáhlosti. Tudíž pro lepší pochopení a snazší analýzu těchto dat vznikla vědní disciplína zvaná *„dolování dat v e-learningových systémech.“*

International educational data mining society (IEDMS), která mimo jiné každoročně organizuje konference týkající se dolování dat v oblasti e-learningu, definuje tuto disciplínu jako: *„Emerging discipline, concerned with developing methods for exploring the unique and increasingly large-scale data that come from educational settings, and using those methods to better understand students, and the settings which they learn in.“* [1].

Ve volném překladu se jedná o proces vyvíjející metody pro zkoumání unikátních a obsáhlých dat, které pochází z výukových prostředí a následné aplikování metod pro lepší pochopení edukace studentů a nastavení obtížnosti výuky (viz. kapitola 2.2).

2.1 Prostředí e-learningových systémů

Základním úkolem e-learningových systémů je vytvoření efektivního prostředí pro řízení a organizaci studia. Tato prostředí sice nemají přesnou definici, ale v obecném měřítku je můžeme identifikovat jako systémy, které poskytují informace týkající se testování probírané studijní látky, kde se neklade důraz na časové omezení nebo polohu přístupu.

Z pohledu studenta můžeme vnímat tato prostředí jako virtuální pracovní prostor, ve kterém se může přihlásit do kurzu, vykonávat testy, stahovat potřebné studijní materiály, reagovat na různé události nebo komunikovat mezi sebou [2].

Pro samotné dolování dat nejsou důležité funkce, ale data, která tyto systémy ukládají. V našem případě se zaměříme na systém eLogika, který je vyvíjen na katedře informatiky Vysoké školy báňské. Systém byl primárně navržen pro výuku matematické logiky. V současné době se kromě matematické logiky v systému vyučují další předměty jako je například matematika pro zpracování znalostí nebo logické programování.

Data, která se v eLogice ukládají a jsou vhodná pro analýzu, pochází z testování studentů nebo z aktivitních logů. Formou logu [3] se zaznamenává průběh online testování nebo přístupy uživatele na webový obsah eLogiky.

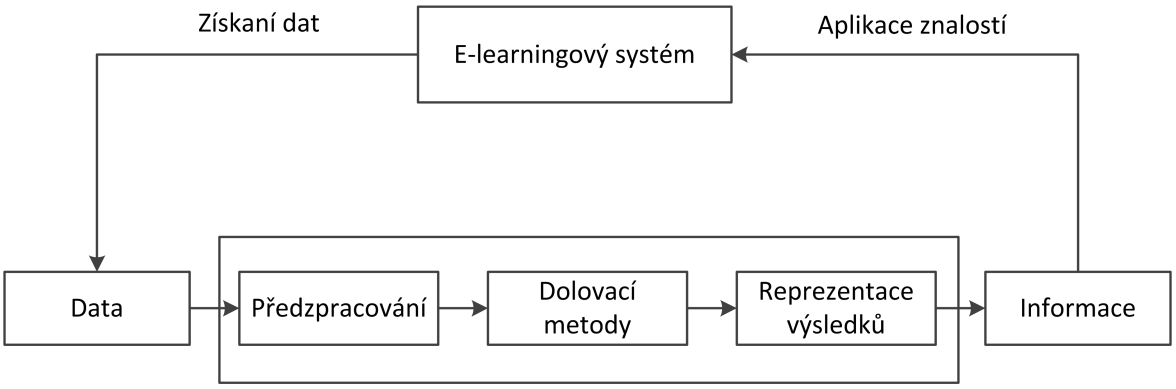
2.2 Proces dolování dat z e-learningu

Proces dolování dat můžeme rozdělit do několika kroků (viz. obrázek 1). V první řadě si musíme stanovit cíle, kterých chceme dosáhnout. Například profilování studentů nebo nastavení obtížnosti výuky.

Po zvolení cílů přichází na řadu předzpracování dat (viz. kapitola 2.2.1), která se upraví do vhodné formy pro vybrané dolovací metody (viz. kapitola 2.2.2).

Nakonec prezentujeme získané výsledky v přehledné formě jako je například graf nebo textová tabulka. Po splnění všech kroků získáme informace, které nám pomohou odpovědět na naše vybrané cíle [4].

Jako příklad celého procesu dolování dat na vybraný e-learningový systém může posloužit studie systému Moodle [5].



Obrázek 1: Proces dolování dat [4]

2.2.1 Předzpracování dat

Úkolem předzpracování je upravit data do přijatelné formy, aby bylo možné jejich následné nasazení ve vybrané dolovací metodě. Tento krok se může brát jako velmi náročný proces, který může mít vliv na kvalitu celého výsledku. Jedním z důvodu tohoto tvrzení je zpracovávání velkého množství atributů, které mohou mít numerickou nebo kategoriální podobu [6]. Pro usnadnění tohoto dílčího kroku existuje mnoho metod, které můžeme na tyto získané data z e-learningových systémů aplikovat [7].

Mezi základní metody pro předzpracovávání patří:

- Filtrace – Podstatou filtrace je, aby data byla formálně správná a konzistentní. Jedním z úkolu filtrace je také hledání chyb v datech, které mohou mít syntaktickou nebo logickou podobu. Dále pomoci filtrování můžeme získat vhodný prvotní náhled na vybraná data díky statistickým atributům (průměr, medián, četnost a další).

- Transformace - Může nastat situace, kdy pro určitou dolovací metodu nebude stačit pouhá filtrace, ale bude potřeba data převést do specifičtějšího tvaru. Pokud bychom vzali například logy četnosti přístupu na jednotlivé stránky v eLogice, tak bychom získali velké množství numerických údajů rozptýlené ve velkém intervalu. Například při použití měření vzdálenosti, která je potřeba u některých shlukovacích metod, by tento rozptyl dat mohl negativně ovlivnit výsledek celé analýzy. Jednou z možností je data kategorizovat a rozdělit dle četnosti, které můžeme převést do základního formátu pomocí normalizace.
- Odvození - V některých případech dolování se hodí z již existujících atributů vytvořit nové. Pokud bychom při analýze logování přístupů vzali v potaz datum, tak tento jednotný atribut se dá rozšířit na atributy jako je denní období, den, měsíc a rok. Na jednu stranu se sice rozšíří datové úložiště, ale na druhou stranu se ušetří čas při spuštění dolovacího algoritmu takovým způsobem, že se nemusí neustále opakovat výpočty nových atributů.

Jak již bylo zmíněno, předzpracování dat je velmi náročnou procedurou, proto je vhodné pro další proces dolování tyto výsledky uložit do datového úložiště, které bude určeno výhradně pro potřeby dolování. Tímto zajistíme efektivnější průběh celého procesu hlavně z hlediska časového zpracování analýzy. Kdybychom totiž museli při každém spuštění analýzy předzpracovávat data, jednalo by se o časově velmi neefektivní záležitost.

2.2.2 Metody pro dolování dat

Dalším krokem po předzpracování dat následuje aplikace vybraných dolovacích algoritmů. Dolování dat v e-learningu je specifické hlavně tím, že pro určitý vzdělávací problém se hodí charakteristická dolovací metoda. Například pokud bychom chtěli zjistit, které otázky dané kategorie jsou při testech nejsložitější a dělají největší problémy, použili bychom některou ze shlukovacích metod.

V tabulce 1 se nachází procentuální zastoupení nejvyužívanějších dolovacích metod, které byly použity v e-learningových systémech. Tato statistika vychází ze všech zveřejněných prací v určitém období. Jak si můžeme povšimnout, největší procentuální zastoupení mají klasifikační metody a shluková analýza [8].

Název skupiny metod	Četnost využití	Proc.využití
Klasifikační metody	102	42.15%
Shlukovací metody	65	26.86%
Regresivní metody	37	15.29%
Ostatní metody	22	8.68%
Asociační pravidla	16	6.61%

Tabulka 1: Výčet využití jednotlivých dolovacích metod v období 2010 - 2014 [8] str.1435

V systému eLogika jsou implementovány takové metody, díky kterým mohou být provedeny kompletní analýzy vyhotovených testů. Pod tím si můžeme představit bodové ohodnocení,

procentuální úspěšnost dílčích otázek a odpovědí. Další náhled na studenty může poskytnout analýza probíhajících kurzů, ať už se jedná o rozdělení témat do kategorií či vytvoření obsahu kurzu [9].

- Shluková analýza

  Popis: Rozdělení objektů do shluků, dle podobnosti.

  Metody: Nehierarchické - K-means, QuickRock. Hierarchické - Metoda nejbližšího souseda.

- Rozhodovací stromy

  Popis: Rozdělení objektů dle informačních atributů do stromové struktury.

  Metody: ID3.

- Asociační pravidla

  Popis: Hledá skryté vztahy mezi atributy.

  Metody: Apriori (algoritmus pro hledání frekventovaných vzorů).

Součástí jsou také statistické metody. Například můžeme zmínit analýzu výsledků testů, ze které se dá určit, jestli se v testech nevyskytují příliš obtížné otázky.

3 Vybrané metody pro dolování dat

V této kapitole jsou popsány dvě vybrané skupiny metod, na které se aplikují data z e-learningového systému eLogika. Nejprve jsou popsány statistické metody, díky kterým se budou statisticky popisovat jednotlivé shluky. Dále je popsána samotná shluková analýza ze které jsou vybrány hierarchické aglomerativní algoritmy (viz. kapitola 3.2).

V druhé části jsou popsána asociační pravidla, kde pro nalezení frekventovaných množin je použit algoritmus Fp-Growth (viz. kapitola 3.3).

3.1 Statistické metody

Statistické metody se dají dobře využít pro analyzování jednotlivých atributů, které byly v dolovacích algoritmech použity. Jak již bylo zmíněno v kapitole předzpracování dat, výsledky těchto výpočtů nám mohou dát prvotní náhled na použitá data.

Mezi vybrané výpočty patří určení maxima, minima a aritmetického průměru, který se dá použít pro výpočet směrodatné odchylky a rozptylu [10].

- Aritmetický průměr $\bar{x}$, je podíl součtu všech hodnot a jejich celkovým počtem.

$$\bar{x} = \frac{1}{n}\sum_{i=1}^{n} x_i$$

- Rozptyl σ^2 je definován jako průměrná kvadratická odchylka mezi údaji souboru $X = [x_1 ... x_n]$ a jejich aritmetickým průměrem $\bar{x}$.

$$\sigma^2 = \frac{1}{n}\sum_{i=1}^{n}(x_i - \bar{x})^2$$

- Směrodatná odchylka σ je definována jako druhá odmocnina z rozptylu.

$$\sigma = \sqrt{\sigma^2} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(x_i - \bar{x})^2}$$

3.2 Shluková analýza

Shluková analýza seskupuje objekty do shluků na základě jejich vzdálenosti. Objekty jsou ve shlucích reprezentovány ve formě n-rozměrných vektorů. Výslednou vzdálenost mezi shluky můžeme vypočítat pomocí mezi-shlukové nebo vnitro-shlukové vzdálenosti (viz. kapitola 3.2.1).

Pro vytváření shluků existují algoritmy, které se dělí na dvě skupiny. První skupina jsou algoritmy nehierarchické. Druhá skupina, která je zde popsána, jsou algoritmy hierarchické [11]. Dále můžeme algoritmy dělit na aglomerativní a divizní (viz. kapitola 3.2.2).

Pro zefektivnění analýzy se používá validační index, které nám určí kvalitu a relevantní počet výsledných shluků (viz. kapitola 3.2.3).

Ukázka celého procesu shlukování je demonstrována na datech, které jsou zapsány v tabulce 2. Řádek této tabulky popisuje studenta a jeho četnost přístupu k webovému obsahu e-learningového systému.

-	P1	P2	P3	P4	P5	P6
S1	0	1	7	3	5	0
S2	4	0	6	1	4	0
S3	0	1	5	3	5	1
S4	2	1	4	0	2	0
S5	3	1	4	0	2	0
S6	1	0	6	0	1	0
S7	0	0	5	0	1	0
S8	0	0	5	0	2	0

Tabulka 2: Přístup studentů k jednotlivým stránkám v rámci e-learningového systému.

3.2.1 Určení vzdálenosti mezi objekty

Pro měření podobnosti shluků bude využita vzdálenost, která je zobrazena v metrickém prostoru pomocí vybrané metriky.

Metrický prostor je dvojice (X, d), kde X je neprázdná množina a d je metrika, která je definována jako zobraznení $d : X \times X \to R$, splňující pravidla pro libovolné $x, y, z \in X$ [12].

$$d \text{ je nezáporná: } d(x, y) \geq 0$$

$$d \text{ je totožná: } d(x, y) = 0 \equiv x = y$$

$$d \text{ je symetrická: } d(x, y) = d(y, x)$$

$$d \text{ splňuje trojúhelnikovou nerovnost: } d(x, y) \leq d(x, z) + d(z, y)$$

Euklidovská metrika přiřazuje dvěma vektorům v euklidovském n-prostoru hodnoty $x = (x_1, \ldots, x_n)$ a $y = (y_1, \ldots, y_n)$, jejichž výsledkem je vzdálenost mezi těmito dvěma vektory $x, y \in R^n$.

$$d(x, y) = \sqrt{\sum_{i=1}^{n} (x_i - y_i)^2}$$

Příklad: Pro výpočet vzdálenosti použijeme data, která jsou zapsána v tabulce 2. Jelikož metrika splňuje symetrii, stačí vypočítat jenom trojúhelníkovou část matice vzdálenosti, která se nachází nad hlavní diagonálou naplněnou nulami.

-	S1	S2	S3	S4	S5	S6	S7	S8
S1	0.0	4.8	2.2	5.6	6.0	5.3	5.5	4.8
S2		0.0	4.9	3.7	3.3	4.4	5.2	4.7
S3			0.0	4.9	5.4	5.4	5.2	4.5
S4				0.0	1.0	2.6	2.6	2.4
S5					0.0	3.2	3.5	3.3
S6						0.0	1.4	1.7
S7							0.0	1.0
S8								0.0

Tabulka 3: Vypočtená vzdálenost pomocí euklidovské metriky

Výpočet vzdálenosti mezi vektory $S1 = \{0, 1, 7, 3, 5, 0\}$ a $S2 = \{4, 0, 6, 1, 4, 0\}$, pomoci euklidovské metriky $d_E(S1, S2) = 4.8$ (viz. tabulka 3).

Manhattanská metrika je vytvořena linearizací euklidovské metriky, což má za následek snížení významu členů s větším rozdílem mezi dílčími souřadnicemi obou vektorů. Následná absolutní hodnota je nezbytná pro zachování kladné výsledné hodnoty vzdálenosti.

$$d(x, y) \quad = \quad \sum_{i=1}^{n} |x_i - y_i|$$

Příklad: Pomocí manhattanské metriky, která se aplikuje na stejné vektorech jako v předešlém příkladu $S1 = \{0, 1, 7, 3, 5, 0\}$ a $S2 = \{4, 0, 6, 1, 4, 0\}$, se vypočte vzdálenost $d_M(S1, S2) = 9$ (viz. tabulka 4).

-	S1	S2	S3	S4	S5	S6	S7	S8
S1	0.0	9.0	3.0	11.0	12.0	10.0	10.0	9.0
S2		0.0	10.0	8.0	7.0	7.0	9.0	8.0
S3			0.0	10.0	11.0	11.0	9.0	8.0
S4				0.0	1.0	5.0	5.0	4.0
S5					0.0	6.0	6.0	5.0
S6						0.0	2.0	3.0
S7							0.0	1.0
S8								0.0

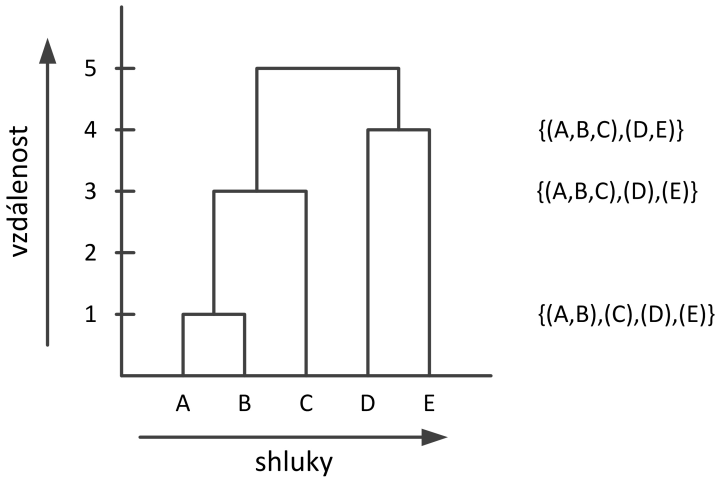
Tabulka 4: Vypočtená vzdálenost pomocí manhattanské metriky

3.2.2 Vybraná skupina algoritmu shlukové analýzy

Vybraná skupina patří do kategorie hierarchického shlukování. Algoritmy se dělí na aglomerativní a divizní. Hlavní rozdíl mezi nimi je v tom, že při aglomerativních metodách se nejprve vytvoří shluky obsahující pouze jeden objekt. Následně postupným slučováním shluků vznikne jeden velký, který bude obsahovat všechny objekty.Oproti tomu divizní algoritmy pracují opačným způsobem.

Další rozdíl mezi těmito algoritmy je ve výpočetní složitosti, kterou algoritmy dosahují při nalezení optimálního výsledku.

Výsledná analýza může být reprezentovaná dendrogramem nebo textovým zápisem (viz. obrázek 2).



Obrázek 2: Znázornění shlukové analýzy dendrogramem a textovým zápisem

Všechny algoritmy pracují rekurzivním způsobem, který začíná na metrickém prostoru (X, d), kde X obsahuje množinu n-objektů $X = \{x_1, \ldots, x_n\}$, pro kterou platí $X \subseteq R^n$. Následně se pomoci metriky vypočte mezi-shluková vzdálenost $d(x_i, x_j)$, která se uloží do matice vzdáleností $n \times n$. Nakonec se porovná vzdálenost podle vybrané metody a vytvoří se hierarchická struktura shluků $\{C_1 \ldots C_n\}$.

Níže jsou popsány vybráne aglomerativní metody, které počítají vzdálenost pomoci metriky d mezi shluky U a V [13].

Akronym	Název metody
sl	Metoda nejbližšího souseda (Single-Linkage)
cl	Metoda nejvzdálenějšího souseda (Complete-Linkage)
avg	Metoda průměrné vazby (Average-Linkage)
cent	Centroidní metoda (Centroid-Linkage)
ward	Wardova metoda

Tabulka 5: Seznam hierarchických aglomerativních metod

Metoda nejbližšího souseda - Má za úkol vytvářet shluky z objektů nebo shluků, které mají mezi sebou nejmenší vzdálenost. Výpočet probíhá tak, že se vezme nejmenší vzdálenost dvou objektů z dvou různých shluků. Velkou nevýhodou je, že pokud jsou vzdálenosti shluku stejné nebo velmi podobné, tak dochází k zřetězení.

$$sl(U,V) = \min_{u \in U, v \in V} d(u,v)$$

Metoda nejvzdálenějšího souseda - Slučuje objekty nebo shluky, které mají vzdálenost od sebe největší. To znamená, že se vezme největší možná vzdálenost, každých dvou objektů z dvou shluků. Z takto vypočtených vzdáleností vezme tu nejkratší a konkrétní shluky se sloučí.

$$cl(U,V) = \max_{u \in U, v \in U} d(u,v)$$

Metoda průměrné vazby - Slučuje shluky, které mají průměr vzdáleností mezi všemi objekty dvou shluků nejmenší.

$$avg(U,V) = \frac{1}{|U| \cdot |V|} \sum_{u \in U, v \in V} d(u,v)$$

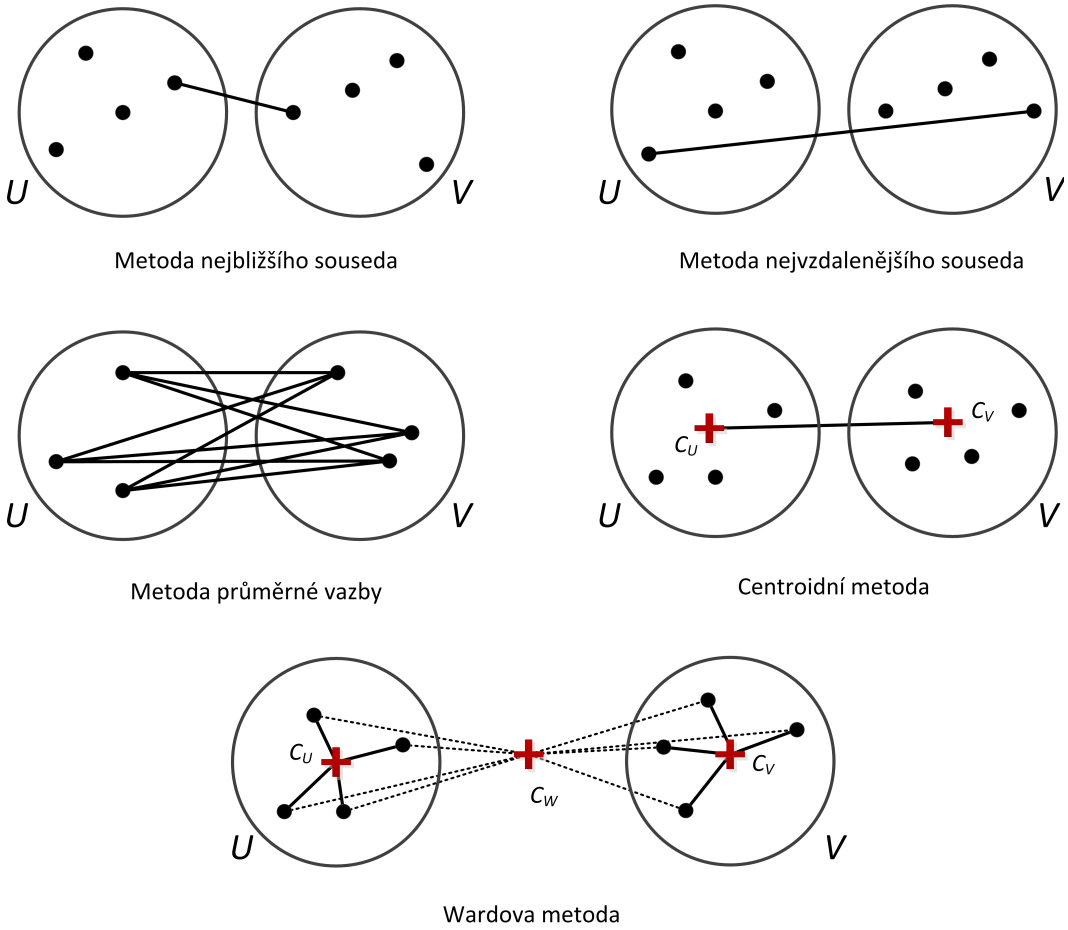
Centroidní metoda - Slučuje shluky na základě jejich těžiště $\mathbf{c}$. Těžiště shluku můžeme vypočítat jako podíl součtu všech objektů ve shluku a velikosti shluku. Následné slučování probíhá tak, že se porovná nejmenší vzdálenost mezi těžišti dvou různých shluků. Po sloučení se vytvoří nové težiště.

$$\mathbf{c}_U = \frac{1}{|U|} \sum_{u \in U} u$$

$$cen(U,V) = d(\mathbf{c}_U, \mathbf{c}_V)$$

Wardova metoda - Je kombinovaný postup, kdy se vzdálenost mezi shluky vypočítá jako přírůstek součtu čtverců odchylek mezi vytvořeným těžištěm $\mathbf{c}_W$ a objekty o_{ij} v obou porovnávaných shlucích, oproti součtu čtvercových odchylek mezi objekty a těžištěm v obou shlucích.

$$ward(U,V) = \sum_{o_{ij} \in U \cup V} d^2(o_{ij}, c_W) - \left(\sum_{o_i \in U} d^2(o_i, c_U) + \sum_{o_j \in V} d^2(o_j, c_V) \right)$$



Obrázek 3: Znázornění hierarchických aglomerativních metod

V literatuře je popsán algoritmus aglomerativního shlukování, který je demonstrován na metodě centroidní a průměrné vazby [14].

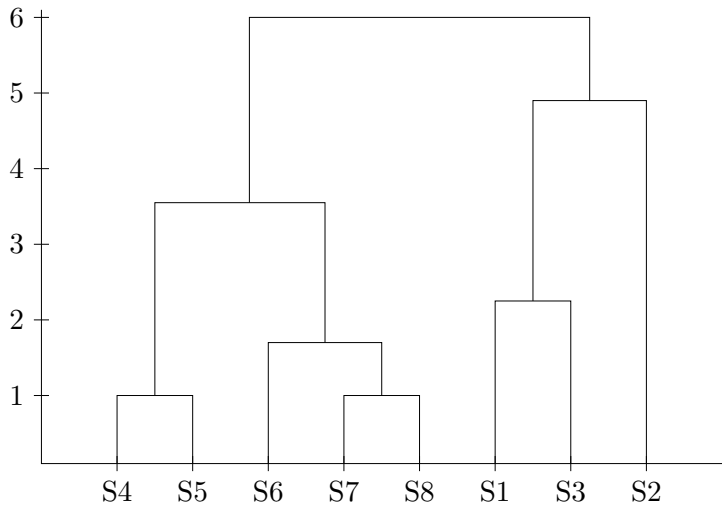
1. Nejprve vypočteme vzdálenostní matici pomocí vybrané metriky.
2. Vytvoří se n shluků, z nichž každý bude obsahovat jeden objekt.
3. Projde se vzdálenostní matice a najdou se dva shluky ($i-t\acute{y}$ a $j-t\acute{y}$), jejichž vzdálenost je nejmenší.
4. Spojením dvou shluků se vytvoří nový $z-t\acute{y}$ shluk. V matici vymažeme $i-t\acute{y}$ a $j-t\acute{y}$ řádek i sloupec. Smazaný řádek i sloupec nahradíme hodnotami $z-t\acute{e}ho$ shluku, které se vypočítají podle stanovené aglomerativní metody. Tímto se vzdálenostní matice sníží o jeden řád.
5. Proces shlukování pokračuje do chvíle, dokud nezbyde pouze jeden velký shluk, který obsahuje všechny objekty. Pokud tomu tak není, přejde se na krok číslo 3.

- Vstupem algoritmu aglomerativního shlukování je pole vektorů reprezentující objekty, metrika pro určení vzdálenosti a vybraná aglomerativní metoda.
- Výstupem je n počet shluků, úroveň, ve které se shluk nachází, jeho těžiště a objekty (pole vektorů), které do něj patří.

```
List<Cluster> clustering(int k, List<Vectors> vector)
{
   Distance.createDistanceMatrix(vector);
   OneCluster(vector);
   List<Cluster> clusters = new List<Cluster>();
   while ((vector.getCount() - clusters.getCount() > k)
   {
      double bestValue = Distance.compareDistanceMatrix();
      clusters.add(Method.mergeCluster(bestValue));
      Distance.recaltulatedDistanceMatrix();
   }
   return clusters;
}
```

Výpis 1: Nástin vytváření shluků v jazyce C#.

Priklad: Pro demonstrativní příklad shlukování použijeme data z tabulky 1 a vzdálenost vypočtenou pomocí euklidovské metriky z tabulky 3. Pro přepočet matice použijeme metodu průměrné vazby. Výsledkem bude dendrogram (viz. obrázek 4).



Obrázek 4: Výsledný dendrogram metody nejvzdálenějšího souseda

3.2.3 Validace shlukové analýzy

Ve shlukování máme několik způsobu, jak rozdělit objekty do shluků. Získaný výsledek nemusí být nejlepší. Pro tyto případy existuje takzvaná validace shlukové analýzy, která nám může vyhodnotit, jak kvalitu shlukování, tak i optimální počet shluků, do kterých jsou objekty rozděleny. Níže jsou popsány dvě validační metody, podle kterých se dá kvalita měřit [15].

Dunnův index - Pro všechny oddělené shluky, kde C_i představuje i- ty shluk, je počítán tento index podle vzorce níže, kde $d(C_i, C_j)$ je minimální mezi-shluková vzdálenost, $d'(C_k)$ je maximální vzdálenost uvnitř shluků a n je počet shluků.

$$d'(C_k) = \sum_{x_{i,j} \in C_k} d(x_i, x_j)$$

$$D = \min_{1 \leq i < j \leq n} \left\{ \frac{d(C_i, C_j)}{\max\limits_{1 \leq i \leq n} d'(C_k)} \right\}$$

Dunnův index je poměr mezi nejmenší mezi-shlukovou a největší vnitro-shlukovou vzdálenosti, který nabývá hodnot nula až nekonečno. Jinými slovy pokud je vzdálenost středů shluku větší, dosahuje index vyšší hodnoty, což nám indikuje lepší rozdělení shluků.

Metoda siluety - Výpočet siluety, pod kterou si můžeme představit poměr podobnosti a odlišnosti mezi shluky, se dělí na tři kroky. Nejprve musíme získat šířku siluety pro každý objekt $S(i)$, dále průměrnou hodnotu siluety pro každý shluk a nakonec celkovou průměrnou šířku siluety pro daný soubor. Šířku siluety pro každý objekt $S(i)$ vypočteme podle vzorce, kde $a(i)$ je průměrná vzdálenost $i-tého$ objektu od ostatních objektů v rámci jednoho shluku a $b(i)$ je minimum z průměru všech vzdáleností $i - tého$ objektu ke všem shlukům.

$$a(i) = \frac{1}{|C_k| - 1} \sum_{x_{i,j} \in C_k} d(x_i, x_j)$$

$$b(i) = \min_{\forall C_k} \left\{ \frac{\sum\limits_{x_i \notin C_k, x_i \in C_k} d(x_i, x_j)}{|C_k|} \right\}$$

$$S(i) = \frac{(b(i) - a(i))}{max\left\{b(i), a(i)\right\}}$$

Metoda siluety může nabývat hodnot od -1 do 1. Pokud se hodnota siluety $S(i)$ blíží k jedné, znamená to, že objekt je zařazen ve správném shluku. Oproti tomu, pokud se hodnota blíží k záporu, objekt by měl být přirazen do jiného shluku. Pro celkové optimální řešení počtu vytvořených shluků se bere vypočtená hodnota, která je nejblíže k jedné.

3.3 Asociační pravidla

Druhou vybranou metodou jsou asociační pravidla. Hledání těchto pravidel můžeme rozdělit na dvě části. První části se generují frekventované množiny, které vytvoříme pomoci algoritmu Fp-growth (viz. kapitola 3.3.1). Tyto množiny se použijí v druhé části pro vytvoření asociačních pravidel (viz. kapitola 3.3.2).

Hlavním úkolem asociačních pravidel je hledat skryté asociace v datových položkách. Tyto položky můžeme definovat jako množinu $I = \{I_1, \ldots, I_k\}$, která obsahuje k vzájemně odlišitelných atributů. Transakce na I je funkce $T : \{1, \ldots, n\} \longrightarrow P(I)$, množina $T(k)$ je $k-t$á transakce z T.

Samotné asociační pravidlo je pak ve formě implikace $X \Rightarrow Y$, kde X značí výchozí předpoklad a Y závěr, pro které platí $X, Y \subseteq I$ a $X \cap Y = \emptyset$.

Při velkém množství transakcí se dá vygenerovat velké množství asociačních pravidel, které pro výslednou analýzu nemusí být relevantní. Pokud bychom chtěli získat silná asociační pravidla, která by byla pro naši analýzu relevantní, je vhodné použít podporu a spolehlivost [16].

- Podpora znamená, jak často se vyskytuje množina položek $X \subseteq I$ v transakcích $T(k)$, pro které platí $X \subseteq T(k)$, kde $T(k) \in T$.

$$podpora(X) \quad = \quad \frac{|\{T(k); X \subseteq T(k)\}|}{n}$$

- Spolehlivost znamená, jak často se vyskytuje množina položek $X \cup Y$ v X transakci $t \in T$ vzhledem k podpoře množiny položek X.

$$spolehlivost(X \Rightarrow Y) \quad = \quad \frac{podpora(X \cup Y)}{podpora(X)}$$

Abychom opravdu mohli říci, že asociační pravidla jsou silná, musí splňovat určenou minimální hranici těchto definovaných hodnot, jak je uvedeno v příkladu níže.

ID	I1	I2	I3	I4	I5
T1	1	1	0	0	1
T2	0	1	0	1	0
T3	0	1	1	0	0
T4	1	1	0	1	0
T5	1	0	1	0	0
T6	0	1	1	0	0
T7	1	0	1	0	0
T8	1	1	1	0	1
T9	1	1	1	0	0

Tabulka 6: Příklad spolehlivosti a podpory

Příklad: V tabulce 6 jsou znázorněny atributy I1 až I5, které nabývají hodnot 0 a 1. Pokud budeme hledat pravidlo $I2 \Rightarrow I1$,tak spočítáme podporu $p = \frac{4}{9} = 0,44 \doteq 44\%$ a spolehlivost $s = \frac{4}{7} = 0,57 \doteq 57\%$. Když budeme mít stanovenou minimální podporu 20 % a minimální spolehlivost 50 %, tak se bude jednat o silné pravidlo.

3.3.1 Vyhledávání frekventovaných množin

Vyhledávání frekventovaných množin tvoří podstatnou část procesu dolování asociačních pravidel. Jedná se o náročnou operaci vytváření takových $\mu-$ frekventovaných množin $\subseteq I, podpora(\mu - fm) \leq \mu$, kde μ nabývá hodnotu $< 0,1 >$.

Z tohohle hlediska by měl vybraný algoritmus splňovat podmínku, co nejmenší časové složitosti, ze které se následně odvíjí efektivnost celého vyhledávání.

Pro tento účel byl vybrán algoritmus zvaný Fp-Growth. Hlavní výhoda tohoto algoritmu tkví v tom, že přistupuje k datům pouze dvakrát. Oproti tomu například algoritmus Apriori, který je v oblasti vyhledávání frekventovaných množin známější, musí přistupovat k datům při každé iteraci. Nevýhoda tohoto algoritmu je v tom, že musí pracovat s větší kapacitou pamětí hlavně při stavbě vyhledávacího stromu, který je jednou z částí celého algoritmu [17].

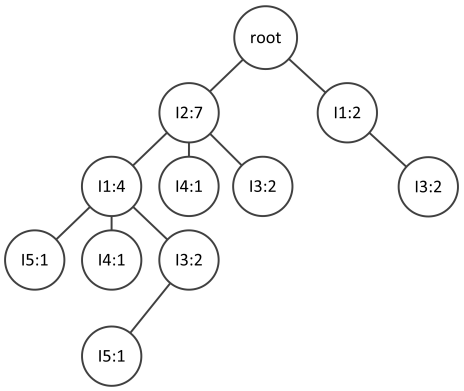
Pro demonstraci algoritmu budou využita data, která se nachází v tabulce 6. Jedná se o devět transakcí obsahující položky I1 až I5. Kromě definovaných dat se ještě určí minimální podpora, kterou pro náš příklad určíme 20 % ($tj. \mu = 0,2$).

Jak bylo zmíněno, algoritmus prochází vybraná data dvakrát. Při prvním průchodu se vytvoří množina L, která bude obsahovat položky a jejich četnosti.

$$L = \{I2:7, I1:6, I3:6, I4:2, I5:2\}$$

Následně se provede druhý průchod, kdy se v každé transakci seřadí položky dle nejčastějšího výskytu v množině L. Důvod toho seřazení je pro následnou stavbu vyhledávacího stromu. Tento strom je tvořen kořenem, na který se napojují jednotlivé položky z transakce jakožto uzly.

Vezmeme první seřazenou transakci $T1 = \{I2, I1, I5\}$ a budeme vkládat jednotlivé položky od kořene stromu. Tímto nám vznikne ve vyhledávácím stromu první cesta. Poté vezmeme další transakci v pořadí tedy $T2 = \{I2, I4\}$ a provedeme stejný postup. Znovu se začne od kořene stromu, jelikož položka $I2$ je již potomek rodiče v tomhle případě kořene, inkrementuje se četnost tohoto uzlu. Pokud by uzel ještě nebyl vytvořen, vznikl by nový s četností jedna. Tento postup se opakuje, dokud nejsou vyčerpány všechny transakce a není vytvořen konečný vyhledávací strom (viz. obrázek 5).



Obrázek 5: Konečný vyhledávací strom

Pro výše popsaný algoritmus vytvoření vyhledávacího stromu je zde uveden kód.

- Vstupem algoritmu je pole vektorů, kde vektor je reprezentován transakcí.
- Výstupem algoritmu je vyhledávací strom.

```
private Tree buildTree(List<Vector> vector)
{
   List<string> orderedTransaction;
   Node root = createRoot();
   finalTree = new Tree(root);
   foreach (List<string> oneTran in vector)
   {
      orderedTransaction = orderOneTransaction(oneTran);
      bool firstNode = true;
      Node parentNode = null;
      createPath(ref orderedTransaction, ref finalTree);
   }
   return finalTree;
}
```

Výpis 2: Nástin vyhledávacího stromu v jazyce C#.

Po vytvoření kompletního vyhledávacího stromu se mohou začít získávat frekventované množiny. Výpočet se skládá z několika kroků, které probíhají pro každou položku neboli sufix v obsažené množině L.

Nejprve vytvoříme podmíněný základ. Tento základ obsahuje množiny uzlů nacházející se mezi sufixem a kořenem stromu. Každá množina je reprezentována svou četností výskytu rovnající se výskytu četnosti sufixu.

Jako příklad vezmeme sufix $I5$, který se nalézá v kompletním vyhledávacím stromu dvakrát. První cesta od sufixu je $< I2, I1, I5 : 1 >$ a druhá je $< I2, I1, I3, I5 : 1 >$. Pokud odstraníme sufix, vzniknou nám dvě množiny $\{I2, I1 : 1\}$ a $\{I2, I1, I3 : 1\}$, které budou tvořit podmíněný základ.

Těmito základy vytvoříme podmíněné vyhledávací stromy. Tyto stromy se vypočítávají podobně jako kompletní vyhledávací strom s tím rozdílem, že se pro stavbu používají množiny, které byly vytvořeny pro podmíněný základ. Důležitou podmínkou je, aby všechny cesty v konečném uzlu splňovali minimální podporu četnosti.

Sufix	Podmíněný základ	podmíněný vyhl.strom	frekventované vzory
I5	{{I2,I1:1},{I2,I1,I3:1}}	<I2:2,I1:2>	{I2,I5:2},{I1,I5:2},{I2,I1,I5:2}
I4	{{I2,I1:1},{I2:1}}	<I2:2>	{I2,I4:2}
I3	{{I2,I1:2},{I2:2},{I1:2}}	<I2:4,I1:2>,<I1:2>	{I2, I3:4}, {I1, I3:4}, {I2, I1, I3:2}
I1	{{I2:4}}	<I2:4>	{I2,I1:2}
I2	{}	<>	{}

Tabulka 7: Generování frekventovaných množin

Když už máme vytvořené podmíněné vyhledávací stromy pro všechny sufixy, nastává poslední krok, a to je generování frekventovaných množin. V principu se jedná o rekurzivní zpracovávání podmíněného vyhledávacího stromu v podobě kombinování všech položek se sufixem. Tímto vznikne konečná množina frekventovaných vzorů (viz. tabulka 7).

3.3.2 Vytvoření asociačních pravidel

Zbývá nám pouze určit silná asociační pravidla pomoci stanovení minimální spolehlivosti. Definice asociačních pravidel byla uvedena v úvodu této kapitoly.

Generování asociačních pravidel je následující vezmeme vygenerovanou $\mu-$ frekventovanou množinu $X = \{I2, I1, I5 : 2\}$ z tabulky 7 a vytvoříme všechny možné podmnožiny S z této množiny ($S \subseteq X$).

$$S = \{\{I1, I2\}, \{I1, I5\}, \{I2, I5\}, \{I1\}, \{I2\}, \{I5\}\}$$

Následně vybereme takové implikace $X \Rightarrow (S - X)$, kdy každá množina splňuje minimální spolehlivost 50 %.

I1,I2	$\Rightarrow$	I5	2/2	100 %
I1,I5	$\Rightarrow$	I2	2/7	29 %
I2,I5	$\Rightarrow$	I1	2/6	33 %
I1	$\Rightarrow$	I2,I5	2/2	100 %
I2	$\Rightarrow$	I1,I5	2/2	100 %
I5	$\Rightarrow$	I1,I2	2/2	50 %

Tabulka 8: Příklad vygenerovaných asociačních pravidel

Závěrem můžeme říci, že minimální spolehlivost 50 %, splňují následující vygenerovaná pravidla: $(I1, I2 \Rightarrow I5; I1 \Rightarrow I2, I5; I2 \Rightarrow I1, I5)$, které dosahují 100 % a jedno pravidlo, které dosahuje spolehlivost 50 % $(I5 \Rightarrow I1, I2)$.

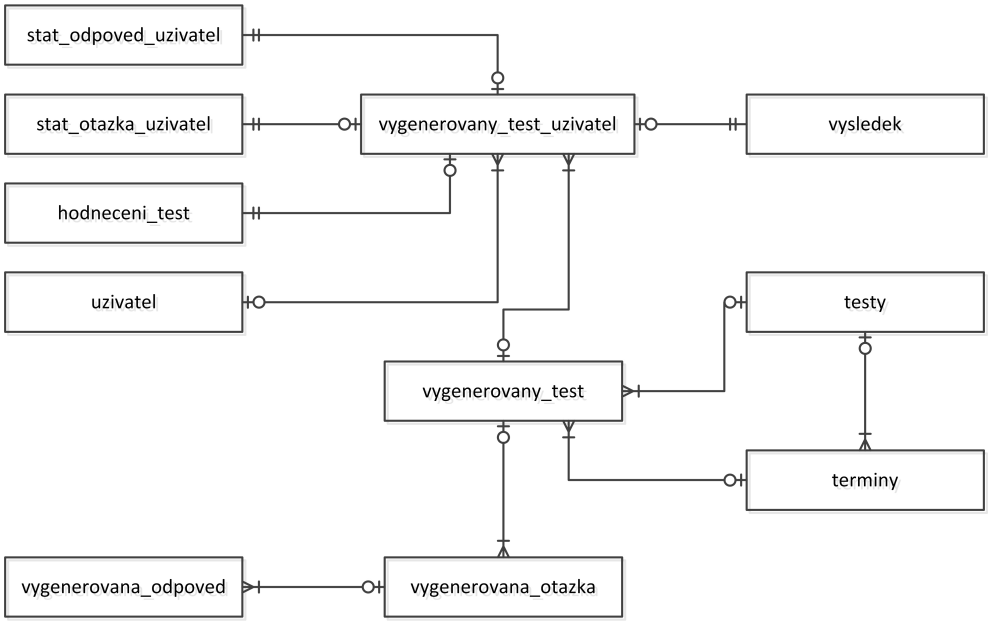
4 Analýza systému eLogika

V této kapitole se zaměříme na vybraný e-learningový systém, nad kterým bude prováděno dolování dat. Nejprve musíme prozkoumat a pochopit datovou strukturu vybraného systému, a poté musíme nalézt data, která jsou pro budoucí analýzu relevantní.

Pro tento účel byl vybrán systém eLogika, který patří do skupiny learning managment systems (LMS) (viz. kapitola 2.1). U tohoto systému můžeme rozdělit zdroj dat na dvě skupiny. První skupina jsou data, která se týkají testování studentů (viz. kapitola 4.1). Druhá skupina dat pochází z logování aktivit uvnitř systému. Může se jednát o přístup na jednotlivé webové stránky nebo chování studenta při probíhajícím online testování (viz. kapitola 4.2).

4.1 Analýza dat z testování

První skupina dat, která se dá analyzovat, pochází z vykonaných testů. Testy mohou být generovány a vyhodnocovány automaticky. Systém uchovává data o vygenerovaných testech pro studenty, kdy každý test má náhodně vygenerované otázky a k nim příslušné odpovědi. Tato data se vytváří v závislosti na kategorii a studijním bloku. Kromě vygenerovaných testů se ukládají také statistická data s vazbou na konkrétního studenta a jeho vypracovaný test. Data se týkají otázek, odpovědí, testů a jejich hodnocení. Menší část tabulek a jejich vazeb je znázorněna v entity relationship diagramu (ER).



Obrázek 6: ER diagram - Část databáze týkající se testování(eLogika)

Pro případnou analýzu se zaměříme na uchovaná statistická data vyhotovených testů a jejich hodnocení.

- **Statistická data vyhotovených testů**

  Statistická data se ukládají do dvou tabulek, jejichž název je *stat_odpoved_uzivatel* a *stat_otazka_uzivatel*. V tabulce *stat_odpoved_uzivatel* je uloženo kromě vazby na test uživatele a zvolenou odpověď také počet kliků. Tabulka *stat_otazka_uzivatel* uchovává data týkající se stráveného času na otázce a počtu zobrazení.

- **Data tykající se hodnocení vyhotovených testů**

  Hodnocení testů, jejichž data jsou relevantní pro případnou analýzu, se ukládá do několika tabulek. Například tabulka *hodnoceni_otazka* uchovává maximální bodové hodnocení otázky a dosažené bodové hodnocení studenta. Tabulka *hodnoceni_test* ukládá maximální bodové ohodnocení testu a také dosažené bodové ohodnocení studenta.

Data obsažena ve zmíněných tabulkách nám mohou dát prvotní náhled o stanovenou obtížnost jednotlivých otázek. Dále můžeme doporučit uživatelům, na které bloky studijní látky by se měli zaměřit, aby si zlepšili výsledky.

4.2 Analýza dat z logovaných aktivit

Druhá skupina dat pochází z logování aktivit. V systému eLogika se na server ukládají aktivita studenta při prohlížení webového obsahu nebo při vykonávání testu. Data jsou sice velmi rozsáhla, ale při správně provedené úpravě se dají dobře analyzovat.

1. **Aktivita přístupu k webovému obsahu**

   Data přístupu jsou uloženy v tabulce *log_access*. Tato tabulka obsahuje několik identifikátoru (škola, kurz, uživatel), které mohou posloužit jako filtr pro výběr určitých dat. Z pohledu užitečných dat, které se dají analyzovat, jsou zde atributy jako je čas, url a ip.

   id_log : Identifikátor logu

   id_uzivatel : Identifikátor uživatele, který aktivitu vykonal

   id_role : Indetifikátor určující roli v systému (Admin,Tajemnik,Garant,Tutor,Student)

   id_skola_info : Indetifikátor školy

   cas : Čas přístupu k webovému obsahu ve formátu 2017-07-17 12:12:01.215

   url : Webová stránka, ke které uživatel přístoupil

   ip : IP adresa, z které byl požadavek na stránku vznešen

   Z těchto dat se dá vytvořit mnoho různých data setů, kterými pak mohou projít různé dolovací algoritmy. Například můžeme analyzovat četnost přístupu k webovému obsahu

v závislosti na denní době nebo ip polohy přístupu. Dále můžeme určit stránky, které jsou pouze „proklikávací“, a to na základě stráveného času, ale také podle toho můžeme navrhnout celou strukturu webu.

2. **Aktivita studenta při online testu**

   Data o chování studenta v průběhu testování jsou uložena v tabulce *onlinetestlogs*. Tato tabulka obsahuje identifikátory (otázka, odpověď, vygenerovaný test), které mohou opět sloužit jako filtr pro výběr dat. Jako užitečná data můžeme brát pořadí zvolené odpovědi, pořadí zvolené otázky, jestli byla odpověď zaškrtnuta a čas, ve kterém byla akce provedena.

   pk_online_test_log : Identifikátor logu

   answerID : Identifikátor zaznamenané odpovědí v testu

   answerOrder : Pořadí, ve kterém byla odpověď vybrána

   questionID : Indetifikátor zobrazené otázky

   questionOrder : Pořadí v jakém byla otázka zobrazena

   userGeneratedTestId : Identifikátor testu uživatele

   isChecked : Indikace zdali byla odpověď zakliknuta

   createdAt : Čas vykonané akce ve formátu 2017-07-17 12:12:01.215

Opět můžeme na základě těchto dat vytvořit mnoho různých datasetů, které se budou zabývat chováním studenta při testování. Navíc můžeme z těchto dat analyzovat i obtížnost testů, otázek nebo celých kategorií. Díky těmto analýzám můžeme také přizpůsobit testovaní, aby bylo efektivnější pro výuku a méně stresující pro studenty ve smyslu nastavení času pro daný test.

5 Analýza pozadavků

Tato kapitola popisuje analýzu a sběr požadavků pro výslednou aplikaci. To znamená, že je popsáno, pro jaký účel je aplikace vytvořena, jakým způsobem se s ní dá pracovat a její vstupy a výstupy (viz. kapitola 5.1). V závěru kapituly je znázorněn případ užití s popsanými jednotlivými případy užití (viz. kapitola 5.2).

5.1 Sběr požadavků

Sběr požadavků je fáze vývoje softwaru. Cílem je definovat požadavky na software a popsat jeho funkčnost. Mezi základní požadavky řadíme otázky typu: k čemu bude aplikace sloužit, kdo s ní bude pracovat, vstupy a výstupy, funkční a nefunkční požadavky.

1. **Účel aplikace**

   Aplikace bude sloužit k analyzování dat z výukového systému eLogika. Mezi hlavní úkoly patří vytvoření funkční aplikace s přehledným uživatelským rozhraním a také vytvořit snadný přístup k datům a jejich znovu použitelnost.

2. **Kdo bude s aplikací pracovat**

   S aplikaci mohou pracovat dva typy uživatelů. První typ je vyučující, který bude mít možnost provádět analýzu kurzu pomocí vybrané dolovací metody. Druhy typ uživatele je developer, který bude mít na starost předzpracování dat a vytváření nových datasetů na základě požadavku učitele.

3. **Vstupy**

   Vstupem aplikace budou data z výukového systému eLogika, která budou zpracována a uložena ve vlastní databázi. Učitel bude mít na výběr kurz a školní rok, ke kterému se přidají další parametry týkající se vybrané dolovací metody. Navíc bude zde možno načíst již dříve vytvořenou analýzu, která bude uložena jako samostatný soubor.

4. **Výstupy**

   Výsledek vybrané analýzy bude přehledně reprezentován v uživatelském rozhraní. Všechny analýzy budou mít přehledný popis použitého datasetu, jeho atributů a nastavených parametrů.

   Shluková analýza: bude mít na výstupu kromě popisu dat, také přehledný graf (dendrogram), který bude obsahovat nastavitelný počet zobrazených shluků. Navíc bude zobrazeno rozdělení shluků pomocí řezu. Každý shluk bude mít také statistický přehled všech obsažených objektů.

   Asociační pravidla: budou mít také na výstupu popis dat. Dále bude na výstupu přehledná tabulka všech asociačních pravidel se specifikovanou podporou a spolehlivostí. Které bude možno seřazovat dle stanovéné podpory nebo spolehlivosti.

5. **Funkční požadavky:**

   - Předzpracování dat z eLogiky
   - Správa datasetů

     Editace datasetů

     Editace atributů pro datasety
   - Shluková analýza

     Editace parametrů

     Zobrazení dendrogramu a popisu dat

     Uložení vykonané analýzy

     Načtení analýzy
   - Asociační pravidla

     Editace parametrů

     Zobrazení popisu dat a vygenerovaných pravidel

     Uložení vykonané analýzy

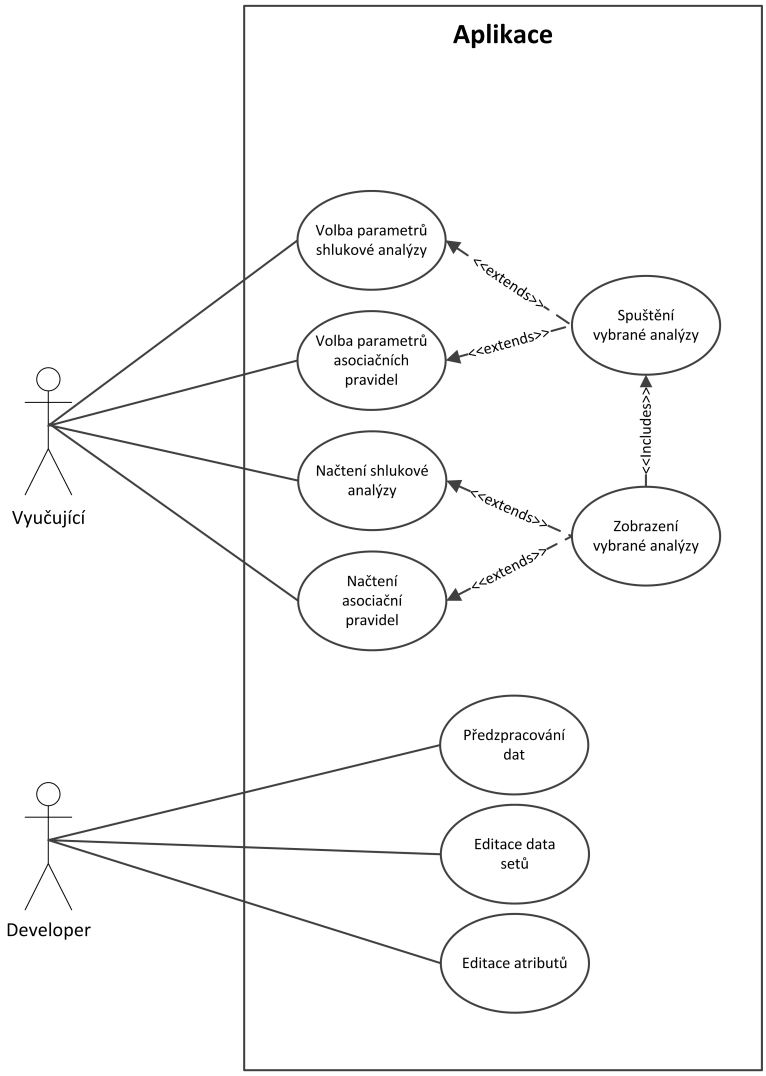
     Načtení analýzy

6. **Nefunkční požadavky:**

   Využitých technologií pro vytvoření aplikace bude hned několik. Uživatel bude mít před sebou desktopovou aplikaci, která bude prezentovat výsledky pomoci windows presentation foundation (WPF). Toto prostředí je postaveno na frameworku .NET 4.5, ve kterém je dostupný jazyk C#. Aplikace bude napsaná již v zmíněném jazyce C#. Výsledné analýzy budou uloženy v Extensible Markup Language (XML) souboru. Pro práci s daty bude vytvořeno datové úložiště MS-SQL, ve kterém budou data uchována.

5.2 Případy užití

Případ užití je posloupnost interakcí, která probíhá mezi aktérem a systémem. Zde je znázorněn a popsán diagram případu užití, který popisuje interakce mezi aktéry (učitel, developer) a systémem (vytvořená aplikace pro dolování dat).



Obrázek 7: ER diagram - Část databáze týkající se testování

5.2.1 Volba parametrů shlukové analýzy

Bude umožněno vybrat shlukovací metodu a metriku.

- Aktéři: Vyučující, systém
- Tok:
  1. Systém zobrazí formulář. Ten bude obsahovat nabídku předmětů, akademických roku pro daný předmět a vytvořené data sety.
  2. Vyučující zvolí předmět, akademický rok a data set a zmáčkne tlačítko načíst data.
  3. Systém načte data a zpřístupní tlačítko spustit analýzu.
  4. Vyučující nastaví shlukovací metodu a metriku. Následně spustí analýzu.
  5. Systém uloží analýzu do XML souboru a zobrazí výsledky uživateli.

5.2.2 Volba parametrů asociačních pravidel

Bude umožněno nastavení podpory a spolehlivosti. Po zobrazení bude moci uživatel vyfiltrovat asociační pravidla na základě podpory a spolehlivosti.

- Aktéři: Vyučující, systém
- Tok:
    1. Systém zobrazí formulář. Ten bude obsahovat nabídku předmětů, akademických roku pro daný předmět a vytvořené data sety.
    2. Vyučující zvolí předmět, akademický rok a data set a zmáčkne tlačítko načíst data.
    3. Systém načte data a zpřístupní tlačítko spustit analýzu.
    4. Vyučující nastaví minimální podporu a spolehlivost. Následně spustí analýzu.
    5. Systém uloží analýzu do XML souboru a zobrazí výsledky uživateli.

5.2.3 Načtení shlukové analýzy nebo asociačních pravidel

V tomto případě bude mít uživatel načíst soubor XML s uloženou analýzou.

- Aktéři: Vyučující, systém
- Tok:
    1. Systém zobrazí formulář. S tlačítkem načíst uloženou analýzu.
    2. Vyučující zvolí načíst uloženou analýzu.
    3. Systém zobrazí uživatele načítaci dialog pro zvolení příslušného souboru.
    4. Vyučující vybere XML soubor ve správném formátu s uloženou shlukovou analýzu.
    5. Systém zobrazí výsledek uložené analýzy.

6 Implementace

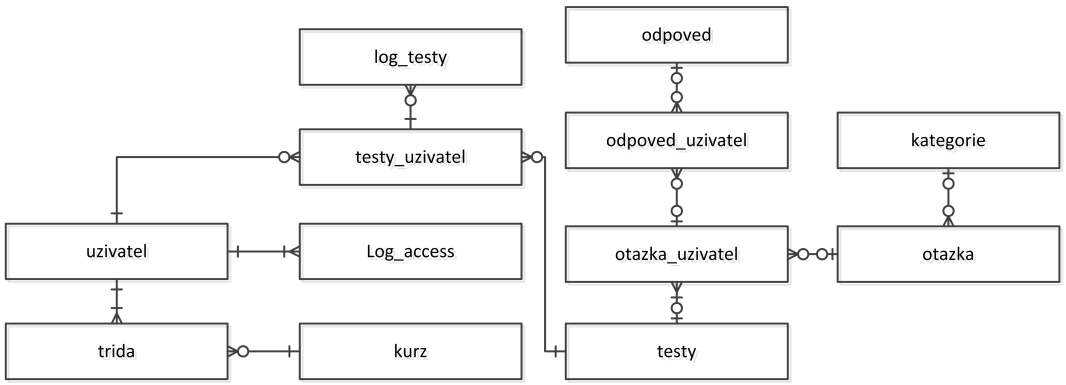
Tato kapitola se věnuje implementaci všech částí aplikace. Nejprve je zde popsána struktura datového úložiště pro uchování předzpracovaných dat (viz. kapitola 6.1). Posléze jsou popsány všechny části výsledné aplikace, které slouží ke zpracování předzpracovaných dat a reprezentování výsledku dolovacích algoritmů (viz. kapitola 6.2).

6.1 Struktura datového úložiště

Datové úložiště bylo vytvořeno pro uchování dat pocházející ze zálohy databáze systému eLogika. Struktura úložiště vychází ze struktury databáze eLogika, kdy je v podstatě úložiště osekáno o nepotřebné tabulky a atributy. Oproti tomu některé tabulky jsou rozšířeny o atributy ulehčující práci s daty.

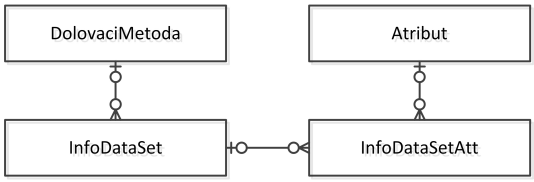
Data, která pochází ze systému eLogika, jsou v neupraveném stavu. Pro správný výsledek dolovacích algoritmů je potřeba tyto data předzpracovat. Pro tento účel bylo napsáno několik uložených procedur pomoci rozšířeného jazyka Transact-Sql (T-SQL), který data kopíruje a zároveň rovnou předzpracovává do vhodného tvaru (viz. kapitola 2.2.1). Kromě zmíněných dat jsou v databázi uložené také informace týkající se vytvořených datasetů.

Výhoda tohoto řešení uložení dat tkví v tom, že při opakovaných analýzách není potřeba opětovného předzpracovávání dat. To nám ušetří celkovou rychlost výsledných analýz. Datové úložiště je znázorněno pomoci ER diagramu (viz. obrázek 8).



Obrázek 8: ER diagram - Část databáze tykající se uložených dat z eLogiky

Níže je znázorněn ER diagram tabulek týkající se uchovávání informací o vytvořených data setech (viz. obrázek 9).



Obrázek 9: ER diagram - Část databáze uchovávající informace o datasetech

6.1.1 Popis tabulek pro předzpracované data

V této podkapitole je napsán stručný popis tabulek, které se nachází v datovém uložišti. K jednotlivým tabulkám je vysvětleno, jakým způsobem se do ní předzpracovávají data a jaké tabulky z eLogiky jsou k tomu potřeba. Podrobný popis jednotlivých atributů nalezneme datovém slovníku, které je součástí přílohy.

- **Tabulka *„uzivatel“***

  Obsah: Obsahuje pouze jediný atribut a to identifikátor uživatele. Tímto je zajištěna anonymita a pro analýzu je tento atribut dostačující.

  Tabulky eLogiky: *uzivatel_role, log_access*

  Předzpracování: Zapsání do tabulky jsou pouze ti uživatelé, kteří mají roli v systému nastavenou jako „student“.

- **Tabulka *„kurz“***

  Obsah: Má uloženy informace o kurzech, které jsou v systému eLogika vytvořeny. Mezi atributy patří identifikátor, název kurzu a akademický rok, ve kterém kurz probíhal.

  Tabulky eLogiky: *kurz_info, kurz, semestr_info, semestr*

  Předzpracování: Byly vybrány všechny kurzy od akademického roku 2013 po současnost. Akademický rok je vypočten z atributů od a do z tabulky semestr.

- **Tabulka *„trida“***

  Obsah: Každému studentovi je přiřazena studijní skupina neboli třída. Kromě identifikátorů třídy a kurzu se také ukládá název, který určuje danou třídu.
  Například ML_EC1_UT_1611 se skládá z názvu kurzu, učebny a dne v týdnu. Dále uchovává získný počet dosáhnutých bodu studentem v dané třídě.

  Předzpracování: Filtrovací parametry jsou dva. Jedná o seznam všech studentů a kurzů, které jsou již v uložišti předzpracovány.

- **Tabulka *„testy“***

  Obsah: V této tabulce jsou uloženy informace o testech, které byly v eLogice vytvořeny. Obsahuje název testu, vymezený čas ke zvládnutí testu, bodové maximum a minimum.

Tabulky eLogiky: *uzivatel_trida, trida_kurz, trida*

Předzpracování: Do úložiště se načtou všechny testy, které mají napojení na konkrétního uživatele.

- **Tabulka *„testy_uzivatele“***

  Obsah: Každému uživateli jsou přiřazeny testy, které v systému vykonal. Kromě identifikátorů je zde zaznamenán časový údaj, který určuje spuštění testu. Dále je zde atribut čas na testu v sekundách, pokus a získané body. Jelikož rozsah bodů se může u každého testu lišit, byl přidán atribut převádějící získané body na procenta.

  Tabulky eLogiky: *vygenerovany_test_uzivatel, vygenerovany_test, termin_student, otazky, otazky_test, stat_otazka_uzivatel, hodnoceni_otazka*

  Předzpracování: Očišťují se atributy s hodnotou null. Dále pomocí uložené procedury s názvem „updateTest“ se vypočítá čas strávený na testu a zisk bodů.

- **Tabulky *„kategorie, otazky ,odpovedi“***

  Obsah: Tabulky mají informativní charakter. Jsou v nich uloženy identifikátory, které mají návaznost mezi kategorii, danou otázkou a její odpovědí.

  Tabulky eLogiky: *vygenerovana_otazka,vygenerovany_test_uzivatel, otazky, stat_otazka_uzivatel, kategorie, vygenerovana_odpoved, odpovedi*

  Předzpracování: Filtrování správných otázek a odpovědí daných kategorii je na základě uložených předzpracovaných testů.

- **Tabulky *„otazka_uzivatel, odpoved_uzivatel“***

  Obsah: Tabulky mají statistkický charakter. Například tabulka *„otazka_uzivatel"* ukládá informace o času stráveném na otázce, počet zobrazení, maximální bodový zisk a získaný bodový zisk. Navíc je zde vytvořen atribut procentuální bodový zisk.

  Tabulky eLogiky: *vygenerovana_odpoved, odpovedi, vygenerovana_otazka, otazky*

  Předzpracování: Kromě klasického kopírovaní dat se převádí čas strávený na otázce do sekund a vypočítává se procentuální bodový zisk.

- **Tabulka *„log_access“***

  Obsah: Obsahuje identifikátor studenta a kurzu. Data vhodné pro analýzu jsou pod atributy URL, čas vytvoření logu a IP přístupu.

  Tabulky eLogiky: *log_access*

  Předzpracování: Byly kategorizovány hodnoty atributu URL v rozsahu P1 až P21, které detekují webový obsah eLogiky (viz. tabulka 9). Podobný postup je proveden u atributu

IP, kde školní IP adresa je nahrazena písmenem „S“ a mimo školní IP adresa je nahrazena písmenem „J“. Toto kategorizování údajů usnadní práci vybírání dat pro případnou analýzu.

- **Tabulka *„log_tests“***

  Obsah: Jedná se o stejnou strukturu jakou má tabulka „onlinetestlog“ nacházející se v eLogice.

  Tabulky eLogiky: *onlinetestlog*

  Předzpracování: Vyberou se pouze logy těch testů, které jsou uloženy v datovém uložišti.

Kategorizace	**Význam URL**
P1	Přihlášení na termín
P2	Vypracování úkolu
P3	Domovská stránka
P4	Vypracované testy
P5	Hodnocení kurzu
P6	Vykonávání aktivity
P7	Rozvrh
P8	Teorie
P9	Vygenerovat test
P10	Konzultační hodiny
P11	Detail rozvrhu
P13	Profil
P14	Přihlášeni na konzultace
P15	Detail teorie
P16	News Longer
P17	Rady
P18	Detail kapitoly
P19	Vyřešené aktivity detail
P20	Vyřešené aktivity hodnocení
P21	Prázdné a neúplné odkazy

Tabulka 9: Seznam kategorizovaných URL

6.1.2 Připravené datasety

Hlavní myšelenka je navrhnout strukturu datasetu pomoci uložené procedury a tu pak uložit v databázi. V databázi se mimo jiné také ukládají informace o vytvořeném datasetu.

V tabulce *„DolovaciMetoda“* se ukládá název skupiny dolovacích algoritmů, jejich popis a indikace, zdali je skupina aktivní. Tabulka *„InfoDataSet“* ukládá název datasetu, název uložené procedury, popis a indikaci, zdali je dataset aktivní. Tabulka *„Atribut“* ukládá informace o použitých atributech v datasetech. Tím se myslí název, popis, typ a indikace, zdali je atribut aktivní. Atribut může být započten do algoritmu nebo může mít status evaluační.

Výhoda tohoto řešení tkví v tom, že se datasety generují přímo v databázi, takže dotaz, který se spouští bývá předkompilován. Tímto se navýší efektivnost a rychlost celé analýzy. Informace, které se ukládají o datasetech, nám více pomohou zpřehlednít vykonávanou analýzu.

Datasetu je možno vytvořit nesčetné množství. Zde závisí na vyučujícím, jaké instrukce předa developerovi.

1. **Analýza logů přístupu v porovnání bodového zisku studenta**

   Dolovací metoda: Shluková analýza

   Popis: Dataset je tvořen atributy četnosti přístupu k webovému obsahu, které nabývají rozmezí P1 až P21. Jako porovnávací atribut slouží bodové hodnocení studenta, kterého dosáhl na konci kurzu.

2. **Analýzu logů přístupu v závisloti na denní době:**

   Dolovací metoda: Shluková analýza

   Popis: Dataset je tvořen atributy četnosti přístupu k webovému jako takovému v závislosti na denní době (dopoledne, odpoledne, večer). Jako porovnávací atribut slouží bodové hodnocení studenta, kterého dosáhl na konci kurzu.

3. **Analýzu logu přístupu dle polohy uživatele:**

   Dolovací metoda: Shluková analýza

   Popis: Dataset je tvořen atributy četnosti přístupu k webovému obsahu, které nabývají rozmezí P1 až P21. Jako porovnávací atributy slouží poloha přístupu dle IP adresy. Tá může být buď školní nebo mimoškolní.

4. **Analýza logu přístupu s indikaci navštívené stránky**

   Dolovací metoda: Asociační pravidla

   Popis: Dataset je tvořen atributy četnosti přístupu k vybranému webovému obsahu. Podle minimální stanovené hodnoty se určí, zdali je stránka navštěvována často nebo zřídka.

```
ALTER PROCEDURE [dbo].[LogyObdobi]
@Kurz varchar(100), @Rok varchar(10)
AS
BEGIN
select distinct
   sum(case when DATEPART(HOUR,Time) > 6 and DATEPART(HOUR,Time) < 12 then 1
       else 0 end) as 'dopoledne',
   sum(case when DATEPART(HOUR,Time) > 12 and DATEPART(HOUR,Time) < 18 then 1
       else 0 end) as 'odpoledne',
   sum(case when DATEPART(HOUR,Time) > 18 and DATEPART(HOUR,Time) < 23 then 1
       else 0 end) as 'vecer',
   t.ziskaneBody from Trida t
join log_Access l on t.id_uzivatel = l.id_uzivatel and t.id_kurz = l.predmet
join Kurz k on k.id_kurz = t.id_kurz
where k.nazev = @Kurz and Akademicky_Rok = @Rok
group by l.id_uzivatel, t.ziskaneBody
END
```

Výpis 3: Uložená procedura - Analýza logů přístupu v závislosti na denní době

6.2 Struktura aplikace

Aplikace je napsána jako třívrstvá architektury, kde jednotlivé vrstvy mezi sebou komunikují. Jedná se o Prezentační, Logickou a Datovou vrstvu. Dále jsou naimplementovány dvě knihovny pro práci s daty. První knihovna s názvem *„DataMiningMethods“* obsahuje dolovací algoritmy. Druhá knihovna, která se stará o reprezentaci výsledků pomocí uživatelského rozhraní nese název *„Vizualization“*. Všechny tyto části jsou podrobněji popsány v podkapitolách této kapitoly.

Knihovna	**Funkce**
DataLayer	Připojení a načtení dat z datového úložiště
BussinessLayer	Zpracovává dotazy z prezentační vrstvy a posílá je do datové vrstvy.
PresentationLayer	Formuláře a navigační komponenty
DataMiningMethods	Dolovací algoritmy
Vizualization	Zpracování vykonaných analýz a prezentování výsledku

Tabulka 10: Seznam naimplementovaných knihoven

6.2.1 DataLayer

Navázání komunikace s databází zajišťuje objekt Connection, kdy přístupové informace k navázání jsou uloženy v souboru s názvem *settings.xml.*

Datová vrstva je napsána pomocí návrhového vzoru Table Data Gateway, kdy jedna vytvořená instance se rovná jedné tabulce. Využívá rozhraní *IGateway*, kde jsou deklarovány základní CRUD operace. Pod těmito operacemi si můžeme představit selektování a upravování dat (Select, Insert, Update, Delete). Objekt uchovávající data určité instance je deklarován ve tvaru *nazevTabulkyObj*. Tyto objekty obsahují deklaraci všech atributů, co se vyskytují v určité databázové tabulce. Objekt *nazevTabulkySQL* obsahuje funkce pro práci s daty. Objekt *nazevTabulkyTable* obsahuje deklaraci všech funkcí CRUD operací a dědí z rozhraní *ITable.*

```
public interface IGateway<T>
{
   Collection<T> Select();
   Collection<T> SelectByParameters(params string[] pamateres);
   int Update(T e);
   int Insert(T e);
   int Delete(int ID);
}
```

Výpis 4: Interface IGateway

6.2.2 BussinessLayer

Tato vrstva se stará o zpracování požadavků z prezentační vrstvy a následně je přeposílá na datovou vrstvu. Například pokud budeme chtít uložit nový data set, tak pošleme data do logické vrstvy, kde bude vytvořen objekt s napojením na datovou vrstvu. Zavolá se metoda *saveDataSet()* a podle parametru, který může nabývat hodnot (INSERT, UPDATE, DELETE) se vybere správná CRUD operace. Pokud bude uložení dat v pořádku, tak se výsledek projeví v prezentační vrstvě. Na této vrstvě se také data mohou vhodně upravovat a přepočítávat.

6.2.3 PrezentationLayer

Na této vrstvě je vytvořeno grafické rozhraní pro určitý typ dolovací metody. Toto rozhraní je vytvořeno pomoci WPF, který je součástí .NET frameworku od verze 3.0. Základním znakem WPF je, že snaží sloučit klasické uživatelské rozhraní využívající komponenty jako je Textbox, Combobox, Datagrid s vytvářením vektorové a rastrové grafiky. Pro popis WPF se používá značkovací jazyk Extensible Application Markup Language (XAML).

```
<Window x:Class="PresentationLayer.Shlukovani"
      xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
      xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
      Title="Shlukovani" Height="496" Width="1200">
<Grid>
   <GroupBox>
      <Canvas>
      </Canvas>
      <TextBox></TextBox>
   </GroupBox>
</Grid>
...
</Window>
```

Výpis 5: Příklad XAML

6.2.4 DataMiningMethods

V této knihovně je naimplementovaná shluková analýza, asociační pravidla, statistické metody a uložení výsledné analýzy do XML souboru.

Hlavní objekt této knihovny je Methods, který obsahuje seznam atributů a vektorů. Dále obsahuje informace o používaném data setu a parametrech pro určitou dolovací metodu. Také se zde vytváří XML soubor, ve kterém se ukládají informace o probíhající analýze. To znamená podrobnosti o datasetu a jeho atributech. V tomto objetu můžeme také vytvořit instanci určité dolovací metody podle nastavených parametrů.

```
<Analyzis>
   <InfoData>
      <Kurz>...</Kurz>
      <Rok>...</Rok>
      <DataSet>...</DataSet>
      <PopisDat>...</PopisDat>
   </InfoData>
   <Attributes>
      <Attribute>
         <Nazev>...</Nazev>
         <Popis>...</Popis>
         <Typ>...</Typ>
      </Attribute>
```

```
<Analyzis>
```

Výpis 6: XML struktura uložení informací o analýze

Shluková analýza je reprezentována objektem Agglomerative. Zde se inicializuje aglomerativní metoda a metrika. Dále obsahuje objekt Cluster, ve kterém je uložena vzdálenost, identifikátor shluku, statistické hodnoty objektů ve shluku, seznam potomků a seznam vektorů. Také obsahuje objekt ValidationObject, ve kterém se nachází seznam validačních indexů k danému počtu shluků. Mezi implementované shlukovací metody patří: Metoda nejbližšího souseda, nejvzdálenějšího souseda, průměrné vazby, centroidní metoda a Wardova metoda. Mezi implementované metriky patří: euklidovská a manhattanská.

```
public void RunClustering()
{
   string distance = parameters["Metrika"];
   string method = parameters["Metoda"];

   Clustering.Hierarchical.Aglomerative aglo = new Clustering.Hierarchical
       .Aglomerative(distance, method, vectors);
   test.generateHierachicalXML(aglo.startAnalyzis());
   test.generatedValidation(aglo.getValidation());
   aglo = null;
}
```

Výpis 7: Spuštění shlukování

Výsledek shlukování se ukládá do vytvořeného XML soubor, kde již jsou uložené informace o vykonávané analýze.

```
<?xml version="1.0"?>
<Analyzis>
   <InfoMetoda>
      <Datum>...</Datum>
      <Metoda>...</Metoda>
      <Metrika>...</Metrika>
      <Objektu>...</Objektu>
   </InfoMetoda>
   <Clusters>
      <Cluster id="" distance="" obj="" idLevy="" idPravy="">
         <Statistics>
            <OneRowStat id="" MAX="" MIN="" AVG="" SUMERR="" SQTRSUM="" />
```

```
        </Statistics>
        <Validations>
          <OneValidation Count="" Dunn="" Siluette="" />
        </Validations>
    </Clusters>
<Analyzis>
```

Výpis 8: XML struktura uložené shlukové analýzy

Asociační pravidla jsou reprezentovány objektem AssociateRules. Tento objekt obsahuje minimální podporu a spolehlivost a také seznam vygenerovaných asociačních pravidel. Každé asociační pravidlo obsahuje identifikátor, výchozí předpoklad, závěr a vypočtenou spolehlivost s danou podporou.

Generování frekventovaných množin je pomocí algoritmu Fp-Growth.

```
public void RunRules()
{
   double support = double.Parse(parameters["Podpora"]);
   double confidence = double.Parse(parameters["Spolehlivost"]);
   AssociateRules.AssociateRules rules = new AssociateRules.AssociateRules(this
       .attributes, this.vectors, support, confidence);
   test.generatedRules(rules.generatedRules());
   rules = null;
}
```

Výpis 9: Spuštění asociačních pravidel

Tak jako v případě shlukové analázy o asociační pravidla mají vytvořenou strukturu XML souboru.

```
<?xml version="1.0"?>
<Analyzis>
   <InfoMetoda>
      <Datum>...</Datum>
      <Spolehlivost>...</Spolehlivost>
      <Podpora>...</Podpora>
      <Transakci>...</Transakci>
   </InfoMetoda>
   <Rules>
      <Rule Ante="" Cons="" Support="" Confidence="" />
   </Rules>
```

```
<Analyzis>
```

Výpis 10: XML struktura uložených asociačních pravidel

6.2.5 Vizualization

Pro reprezentování výsledků se využívá naimplementovaná knihovna zvaná Vizualization. Tato knihovna musí být volána přímo na prezentační vrstvě. O vše se stará objekt *Vizualization*, který načte cestu k souboru vybrané uložené analýzy. Následně vytvoří instanci, která zaručí rychlé a efektivní parsování XML souboru.

```
public IEnumerable<XElement> SimpleStreamAxis(string inputUrl,
string elementName)
{
  using (XmlReader reader = XmlReader.Create(inputUrl))
  {
     reader.MoveToContent();
     while (reader.Read())
     {
        if (reader.NodeType == XmlNodeType.Element)
        {
           if (reader.Name == elementName)
           {
              XElement el = XNode.ReadFrom(reader) as XElement;
              if (el != null)
              {
                 yield return el;
              }
           }
        }
     }
  }
}
```

Výpis 11: Parsování XML souboru

Pokud uživatel načte soubor s uloženou shlukovou analýzou, tak se vytvoří objekt Dendrogram. Tento objekt má jako vstupní parametr canvas, který zajistí vykreslení a následnou editaci výsledného celého dendrogramu.

Nejprve se načtou všechny shluky. Poté podle stanového parametru se vykreslí určitý počet shluků. Každý shluk je tvořen objektem *PointCluster*, který uchovává identifikátor, souřadnice

X,Y a objekt *Path* zaručující spojení všech hran do jednoho geometrického útvaru. To nám zaručí zvýraznění určitého shluku.

```
private void calculateCords(ref PointCluster clusterPoint, int level)
{
   double xCords = (this.layoutWidth / this.countObjects) * level;
   double yCords = this.layoutHeight;
   clusterPoint.setCords(xCords, yCords);
   this.DrawText(clusterPoint.getId().ToString(),xCords+12,yCords,8);
}
```

Výpis 12: Výpočet souřadníc pro základní hladinu shluků dendrogramu

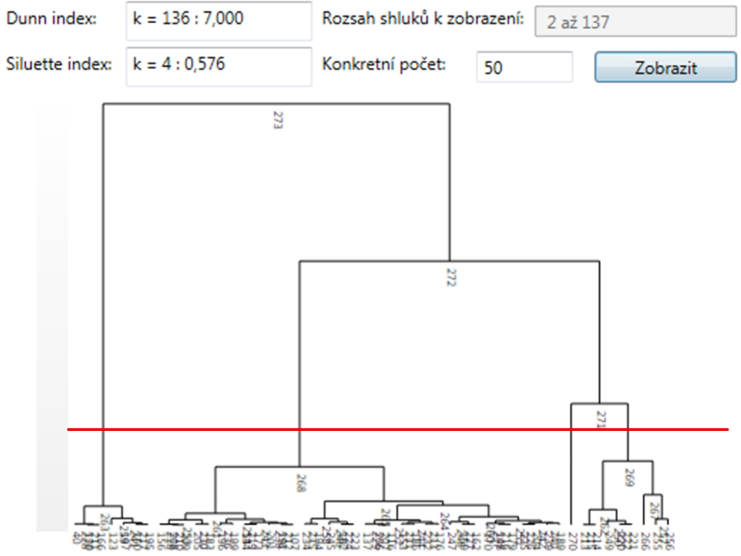
Jestliže uživatel načte soubor s uloženými asociačními pravidly, tak se mu zobrazí tabulka všech vygenerovaných pravidel. V té pomocí filtru spolehlivosti a podpory bude možno zobrazit pouze určitá asociační pravidla. Vše se provádí pomoci metody sortRules, která má na vstupu paremetry: *typ, spolehlivost, podporu a DataGrid.*

7 Experimety s daty

V této kapitole jsou vybrány pro ukázku dva datasety, na které budou aplikovány dolovací metody. Pro shlukovou analýzu jsem vybral analýzu logů přístupu v porovnání bodového zisku studenta. Pro asociační pravidla jsem vybral analýzu logů přístupu s indikací navštívené stránky (viz. kapitola 6.1).

7.1 Experiment pomocí shlukování

Kurz, který byl pro analýzu vybrán, se nazývá matematika pro zpracování znalostí (MPZZ). Akademický rok byl nastaven na 2016/2017. Pro analýzu jsme vybrali wardovu metodu, která využívá čtvercovou euklidovskou metriku. Po spuštění analýzy se vygenerovalo 137 objektů. Každý objekt je reprezentován studentem a jeho přístupu k webovému obsahu. Validační index siluet dosáhl nejlepší hodnoty pro čtyři shluky, která činila 0,576. Pro lepší reprezentaci výsledku pomocí dendrogramu bylo vybráno prvních padesát shluků (viz. obrázek 10).



Obrázek 10: Výsledný dendrogram wardovy shlukovací metody

Wardova metoda se snaží vytvářet shluky stejné velikost. Toto tvrzení, jak můžeme vidět na obrázku, se potvrdilo. Jedná se o shluky číslo 263, 268, 269 a 270. V tabulce 11 je popsán podrobný statistický přehled shluků, ze kterého lze vyvodit aktivitu studentů na webovém obsahu v porovnáním s dosaženými výsledky.

ID shluku	Počet objektů	Průměrný přístup k obsahu	Průměrný zisk bodů
263	54	8 na stránku	17,78 b
268	74	25 na stránku	63,24 b
269	8	61 na stránku	70,25 b
270	1	98 na stránku	65 b

Tabulka 11: Seznam vygenerovaných shluků

Z tabulky je patrné, že v systému je jeden student, jehož aktivita je v průměru větší než u ostatních studentů. Dále vidíme, že ve shluku 269 se nalezla skupinka studentu, jejichž aktivita je vyšší a to se projevilo na jejich studijních výkonech. Podle shluku 263 lze naopak vypozorovat, že skupina s největším počtem studentů má aktivitu na těchto stránkách minimální. Tato skutečnost se projevuje na jejich bodovém zisku. Průměrný bodový zisk dosahuje 17 bodů. Touto analýzou můžeme vybrané skupině studentu doporučit větší aktivitu. Po důkladnější analýze statistických hodnot jednotlivých webových obsahů můžeme zjistit, že jejich aktivita ohledně studování teorie je minimální.

7.2 Experiment pomocí asociačních pravidel

Opět vybereme kurz MPZZ, s tím rozdílem, že tentokrát nastavím akademický rok na 2015/2016. Minimální podpora je nastavena na hodnotu 0.6 a minimální spolehlivost nabývá hodnoty 0.8. Po spuštění analýzy se načetlo 137 transakcí. Z těchto transakcí se vygenerovalo 6192 pravidel. Z hlediska prozkoumávání výsledků je to velmi vysoké číslo. Na tento problém použijeme filtr minimální podpory a spolehlivosti. Nastavíme hodnotu podpory na 90 % a hodnotu spolehlivosti na 95 %. Tímto se z 6192 pravidel zobrazí pouze 100. Následně můžeme tabulku setřídit podle podpory nebo spolehlivosti. Pro přehlednost přepíšeme některé zajímavé pravidla do tabulky, kde nahradíme hodnotu P její URL hodnotou z tabulky 9.

Předpoklad		Závěr	Podpora	Spolehlivost
Teorie	=>	Přihlášení na termín	100%	94,70 %
Rozvrh	=>	Příhlášení na termín	96,92%	99,32 %
Příhlášení na termín	=>	Rozvrh	96,92%	97,33 %
Rozvrh, Příhlášení na termín	=>	Detail rozvrhu	95,36%	98,63 %
Domovská stránka	=>	Rozvrh, Detail rozvrhu	92,05%	99,29 %
Domovská stránka, Teorie	=>	Přihlášení na termín,Rozvrh	92,72%	97,22 %

Tabulka 12: Seznam naimplementovaných knihoven

Z výsledné analýzy a prozkoumání asociačních pravidel můžeme například doporučit developerovi, jak sestavit výsledný web. Dále můžeme z analýzy zjistit, jak studenti pracují v systému eLogika.

8 Závěr

Na začátku této práce je popsán proces dolování dat v oblasti e-learningu a seznámení se s vybraným e-learningovým systémem.

Dále jsou popsány a naimplementovány některé dolovací metody, které se dají použít pro dolování dat ze systému eLogika. Mezi hlavní metody, které jsou popsány, patří shlukování a asociační pravidla. Navíc se k tomu také přidávají základní statistické metody.

Jedním z úkolu této práce bylo prozkoumat uložená data vybraného e-learnigový systému a určit vhodná data pro případnou analýzu pomocí naimplementovaných dolovacích metod. V tomto kontextu bylo navrženo datové uložiště, které slouží výhradně pro potřeby dolování dat.

Co se týče přehledné reprezentace výsledku, tak bylo naimplementováno vlastní generování dendrogramu s označováním vybraných shluků.

Výsledkem je aplikace, která má schopnost tvořit vlastní datasety podle pokynů vyučujícího, pro analyzování dat pocházející z výukového systému eLogika. Díky přehledné reprezentaci výsledků nabízí náhled na chování studentů uvnitř systému.

Tato aplikace i datové úložiště pro dolování se dá v budoucnu rozšiřovat o další metody nebo datasety. Z pohledu stráveného času na vývoji této aplikace a všech jejích částí soudím, že bakalářská práce byla pro mě velmi náročná. Mezi nejnáročnější části práce bych vyzdvihl implementaci generování frekventovaných množin Fp-Growth, implementací vlastního dendrogramu a vytvoření datového uložiště. Na druhou stranu mi dodala spoustu nových znalostí, které jsem musel při vývoji této aplikace nabýt. Ať se jedná o teoretické chápání všech dolovacích algoritmů až po praktickou část práce.

Literatura

[1] International Educational Data Mining Society. [online]. [cit. 2016-03-27]. *Dostupné z: http://www.educationaldatamining.org/* .

[2] What is an LMS?. Mindflash. [online]. [cit. 2016-03-27]. *Dostupné z: https://www.mindflash.com/lms* .

[3] BLACK, Erik W., Kara DAWSON a Jason PRIEM. Data for free: Using LMS activity logs to measure community in online courses. *The Internet and Higher Education 2008* 11(2), 65-70 DOI: 10.1016/j.iheduc.2008.03.002.

[4] ROMERO Cristobal a Ventura Sebastian. Data mining in education. *WIREs Data Mining Knowl Discov* 2013, 3: 12–27 doi: 10.1002/widm.1075

[5] ROMERO Cristobal, Ventura Sebastian, Salcines. E. Data mining in course management systems: Moodle case study and tutorial. *Computer and Education* 2008, 51(1), 368-384.

[6] PEÑA-AYALA, Alejandro (ed.). Data mining in course *Educational data mining: applications and trends.* Cham: Springer, c2014. Studies in computational intelligence, vol. 524. ISBN 978-3-319-02738-8.

[7] ŠARMANOVÁ, Jana. *Metody analýzy dat: učební text.* [CD-ROM]. Vyd. 1. Ostrava: Vysoká škola báňská - Technická univerzita Ostrava, 2012. ISBN 978-80-248-2565-6.

[8] PEÑA-AYALA, Alejandro. *Educational data mining: A survey and a data mining-based analysis of recent works.* Expert Systems with Applications 2014, 41(4), 1432-1462 DOI: 10.1016/j.eswa.2013.08.042.

[9] GERLICH, Jan a MENŠÍK, Marek. Data mining v e-learningu. 2015 [online]. [cit. 2016-03-27] *Dostupné z https://elogika.vsb.cz/workshop/data_a_informace2015/prezentace/prezentace1.pdf*

[10] Matematika. Matematika – Statistika. [online]. 10.6.2016 [cit. 2016-06-10]. *Dostupné z: matematika.cz*

[11] ŠILHAN, David a KELBEL, Jan Shluková analýza [online]. [cit. 2016-03-29] *Dostupné z http://cmp.felk.cvut.cz/cmp/courses/recognition/zapis_prednasky/zapis_02/*

[12] DEZA, M. a Elena. DEZA. *Encyclopedia of distances.* New York: Springer Verlag, c2009. ISBN 978-3-642-00233-5.

London: Springer, c2008. ISBN 9781848002012.

[13] SIMOVICI, Dan A. a Chabane. DJERABA. *Mathematical tools for data mining: set theory, partial orders, combinatorics.* London: Springer, c2008. ISBN 9781848002012.

[14] BOUGUETTAYA, Athman, Qi YU, Xumin LIU, Xiangmin ZHOU a Andy SONG. *Efficient agglomerative hierarchical clustering.* Expert Systems with Applications 2015, 42(5), 2785-2797 DOI: 10.1016/j.eswa.2014.09.054. ISSN 09574174.

[15] Desgraupes, Bernard *Clustering Indices* [online]. [cit. 2017-05-29] University Paris Ouest Lab Modal'X *Dostupne z: https://cran.r-project.org/web/packages/clusterCrit/vignettes/clusterCrit.pdf*

[16] QIANKUM Zhao,BHOWMICK S. Sourav. *Association Rule Mining: A Survey* Technical Report, CAIS, Nanyang Technological University, Singapore, No. 2003116 , 2003.

[17] Garg, Ritu, and Preeti Gulia. "Comparative Study of Frequent Itemset Mining Algorithms Apriori and FP Growth."International Journal of Computer Applications IJCA 126.4 (2015): 8-12. Web.

A Datový slovník

DolovaciMetoda	Typ	Velikost	PK	Null	Popis
IdMetoda	int	-	1	0	ID dolovací metody
Nazev	Varchar	100	0	0	Název skupiny metod
Popis	Varchar	max	0	1	Popis skupiny metod

InfoData	Typ	Velikost	PK	Null	Popis
IdInfo	int	-	1	0	Id data setu
IdMetoda	int	-	1	0	Id dolovací metody
Nazev	Varchar	100	0	0	Název datasetu
NazevProcedury	Varchar	Max	0	0	Název uložené procedury
Aktivni	bit	-	0	0	Zdali je aktivní
Popis	Varchar	max	0	1	Popis skupiny metod

InfoDataAtt	Typ	Velikost	PK	Null	Popis
IdSpoj	int	-	1	0	ID metoda - atribut
IdInfo	int	-	1	0	ID datasetu
IdAtt	int	-	1	0	ID atributu

Kategorie	Typ	Velikost	PK	Null	Popis
IdKategorie	int	-	1	0	pův. Id kategorie
Nazev	Varchar	100	0	0	Název kategorie

Kurz	Typ	Velikost	PK	Null	Popis
IdKurz	int	-	1	0	pův.Id kurzu
Nazev	Varchar	50	0	0	Název kurzu
AkademickyRok	Varchar	50	0	0	Aka.rok

LogAccess	Typ	Velikost	PK	Null	Popis
IdLog	int	-	1	0	pův.Id logu
IdUzivatel	int	-	1	0	pův.Id uzivatele
IdKurz	int	-	1	0	pův.Id kurzu
Time	datetime	-	1	0	čas logu
URL	varchar	100	1	1	Webová adresa
IP	varchar	40	1	1	IP adresa

LogTests	Typ	Velikost	PK	Null	Popis
IdTLog	int	-	1	0	pův.Id logu
IdTU	int	-	1	0	pův.Id testu uzivatele
answerID	int	-	1	0	pův.Id odpovědi
answerOrder	int	-	1	0	pořadí odpovědi
questionID	int	-	1	0	pův.Id otázky
questionOrder	int	-	1	0	pořadí otázky
isCheck	bit	-	1	0	jeli zakliknutá
created	datetime	-	1	0	čas akce

Odpoveď	Typ	Velikost	PK	Null	Popis
IdOdpoved	int	-	1	0	pův.Id odpovedi
správnost	bit		0	0	správnost odpovědi
casReseni	Varchar	50	0	0	cas zaklinutí

OdpoveďUzivatele	Typ	Velikost	PK	Null	Popis
IdUOd	int	-	1	0	pův.Id odpoved uzivatele
IdUO	int	-	1	0	pův.Id otazka uzivatel
IdOdpoved	int	-	1	1	pův.Id odpovedi

OtazkaUzivatel	Typ	Velikost	PK	Null	Popis
IdUO	int	-	1	0	pův.Id otazka uzivatel
IdTU	int	-	1	0	pův.Id test uzivatel
IdTest	int	-	1	0	pův.Id testu
IdOtazka	int	-	1	0	pův.Id otazky
casOtazka	time	-	1	1	cast stráveny na otázce
pocetZobrazeni	int	-	1	1	Počet zobrazení otázky
bodyZisk	float	-	1	0	Získané body
bodyZiskProc	int	-	1	0	Získané body v procentech
maxBody	float	-	1	0	Maximum bodů z otázky

Otazky	Typ	Velikost	PK	Null	Popis
IdOtazka	int	-	1	0	pův.Id otazka
IdKategorie	int	-	1	0	pův.Id kategorie

Testy	Typ	Velikost	PK	Null	Popis
IdTest	int	-	1	0	pův.Id testu
Zarazeni	int	-	1	0	Zařazení testu
Nazev	varchar	50	1	1	Název testu
Cas	int	-	1	0	Limit testu
MaxBody	int	-	1	1	Maximum k získaní
MinBody	int	-	1	1	Minimum k získání

Testy	Typ	Velikost	PK	Null	Popis
IdTU	int	-	1	0	pův.Id test uživatele
IdUzivatel	int	-	1	0	pův.Id uživatele
IdTest	varchar	50	1	0	pův.Id testu
CasSpusteni	datetime	-	1	0	Čas spuštění
Pokus	int	-	1	1	Vykonávaný pokus
CasNaTestu	int	-	1	1	Čas strávený na testu
Body	int	-	1	1	Získané body
BodyProc	int	-	1	1	Body v proccentech

Trida	Typ	Velikost	PK	Null	Popis
IdTrida	int	-	1	0	Id třídy
IdPuvTrida	int	-	1	0	pův.Id třídy
IdKurz	int	-	1	0	pův.Id kurzu
IdUzivatel	int	-	1	0	pův.Id uzivatele
Nazev	varchar	50	1	1	Název třídy
ZiskaneBody	int	-	1	0	Získané body v dané třídě uživatelem

Uzivatel	Typ	Velikost	PK	Null	Popis
IdUzivatel	int	-	1	0	Id uživatele

VybranaAtributy	Typ	Velikost	PK	Null	Popis
IdAtt	int	-	1	0	pův.Id Atributu
Nazev	varchar	30	1	0	Název atributu
Popis	varchar	max	1	1	Popis atribut
Typ	bit	-	1	0	Typ atributu

B Uživatelská příručka

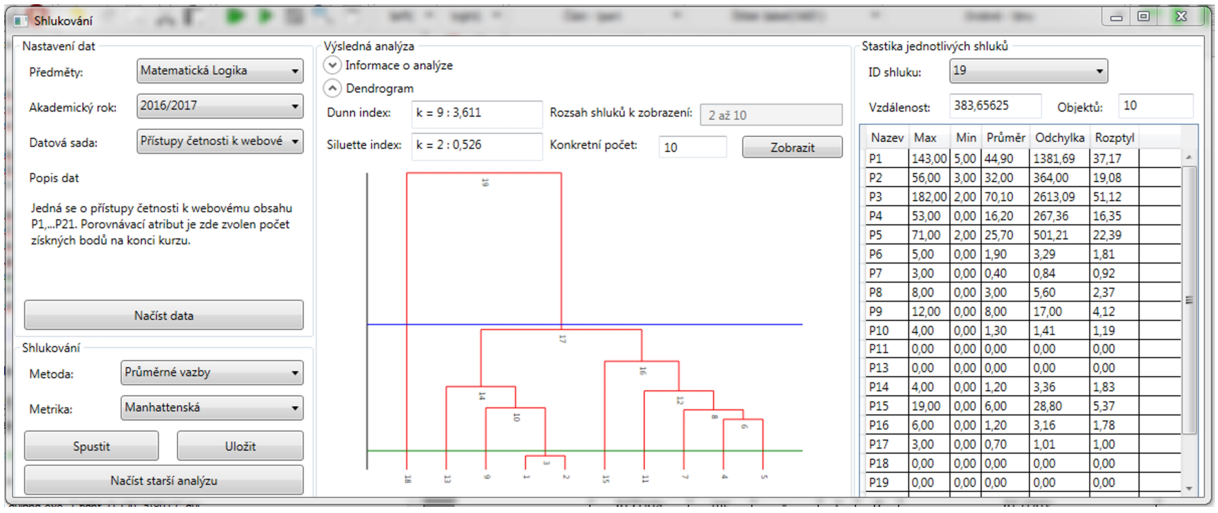
B.1 Shluková analýza

1. Navigační panel

   Tento panel obsahuje navigační tlačítka. Tlačítko „Načíst data“ slouží k načtení dat pomoci vybraných parametrů. Parametry jsou: Kurz, Akademický rok a dataset. Každý vybraný dataset má konkrétní popis využití. Jakmile jsou data načteny, může se provést shlukovací analýza. Na výběr budou shlukovací metody a vybraná metrika. Shluková analýza se spustí po zmáčknutí tlačítka „Spustit“. Výslednou analýzu je také možné uložit a znovu načíst.

2. Výsledek analýzy

   Výslednou analýzu reprezentuje informativní popis datasetu a jeho použitých atributů, ale také popis nastavených parametrů tykající se shlukování. Dále lze vidět přehledný dendrogram se stastistickým přehledem určitého shluku, který lze označit.



Obrázek 11: Uživatelské rozhraní - Shluková analýza

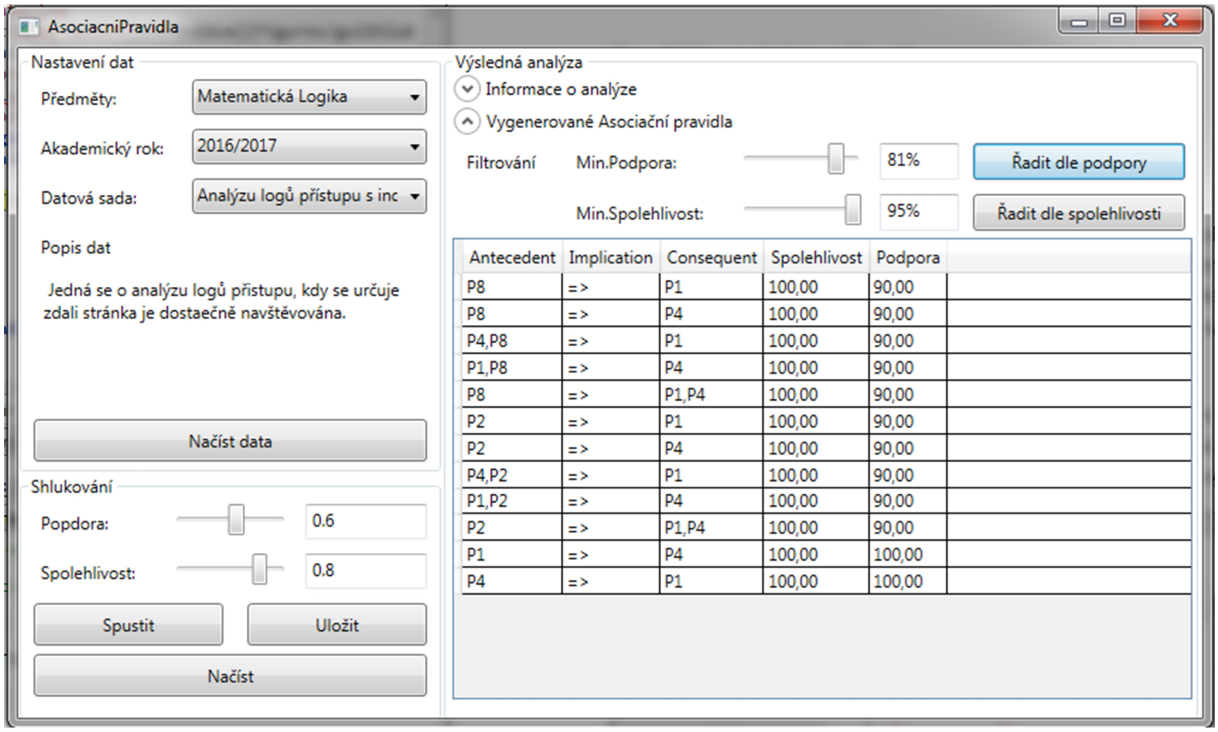
B.2 Asociační pravidla

1. Navigační panel

   Tento panel obsahuje navigační tlačítka. Tlačítko „Načíst data“ slouží k načtení dat pomoci vybraných parametrů. Parametry jsou: Kurz, Akademický rok a dataset. Každý vybraný dataset má konkrétní popis využití. Jakmile jsou data načteny, může se provést generování asociačních pravidel. Paremetry asociačních pravidel jsou minimální podpora a spolehlivost. Generování asociačních pravidel se spustí po zmáčknutí tlačítka „Spustit“. Výslednou analýzu je také možné uložit a znovu načíst.

2. Výsledek analýzy

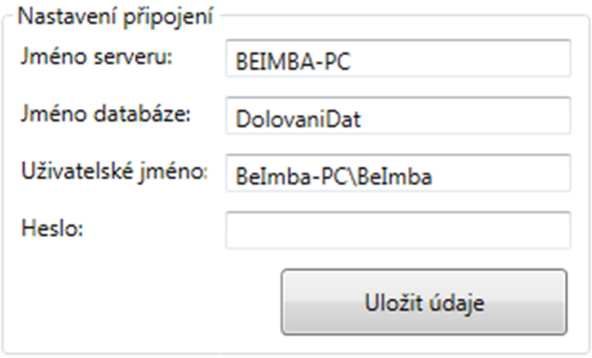
   Výslednou analýzu reprezentuje informativní popis datasetu a jeho použitých atributů, ale také popis nastavených parametrů tykající se asociačních pravidel. Dále lze vidět tabulku, kterou lze vyfiltrovar dle stanové podpory a spolehlivosti



Obrázek 12: Uživatelské rozhraní - Asociační pravidla

B.3 Nastavení pro developery

1. Připojení k databázi Připojení k databázi je následující. Z přílohy se vezme záloha a nahraje na lokální server. Uživatel poté vyplní údaje a dá uložit.



Obrázek 13: Připojení k databázi

C Obsah CD

Přiložené CD obsahuje tyto složky:

- Aplikace - obsahuje projekt aplikace ve Visual Studiu 2013.
- DatoveUloziste - obsahuje zálohy databáze pro dolování dat (MSSQL 2014).
- UlozeneAnalyzy - obsahuje několik už vyřešených analýz.
- Bakalářská práce - obsahuje text diplomové práce ve formátu .pdf